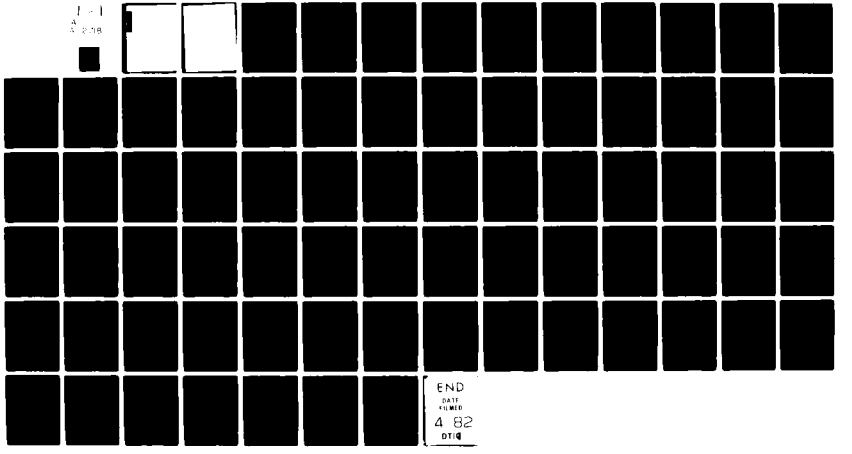
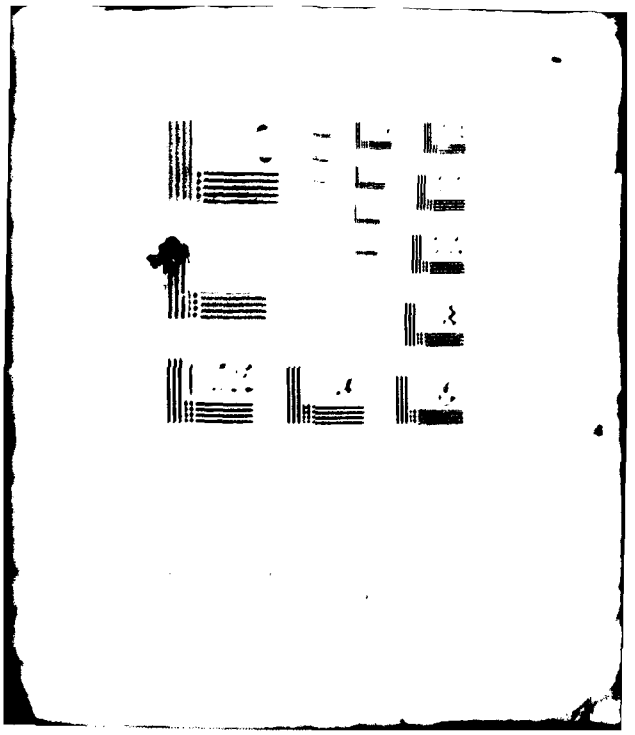
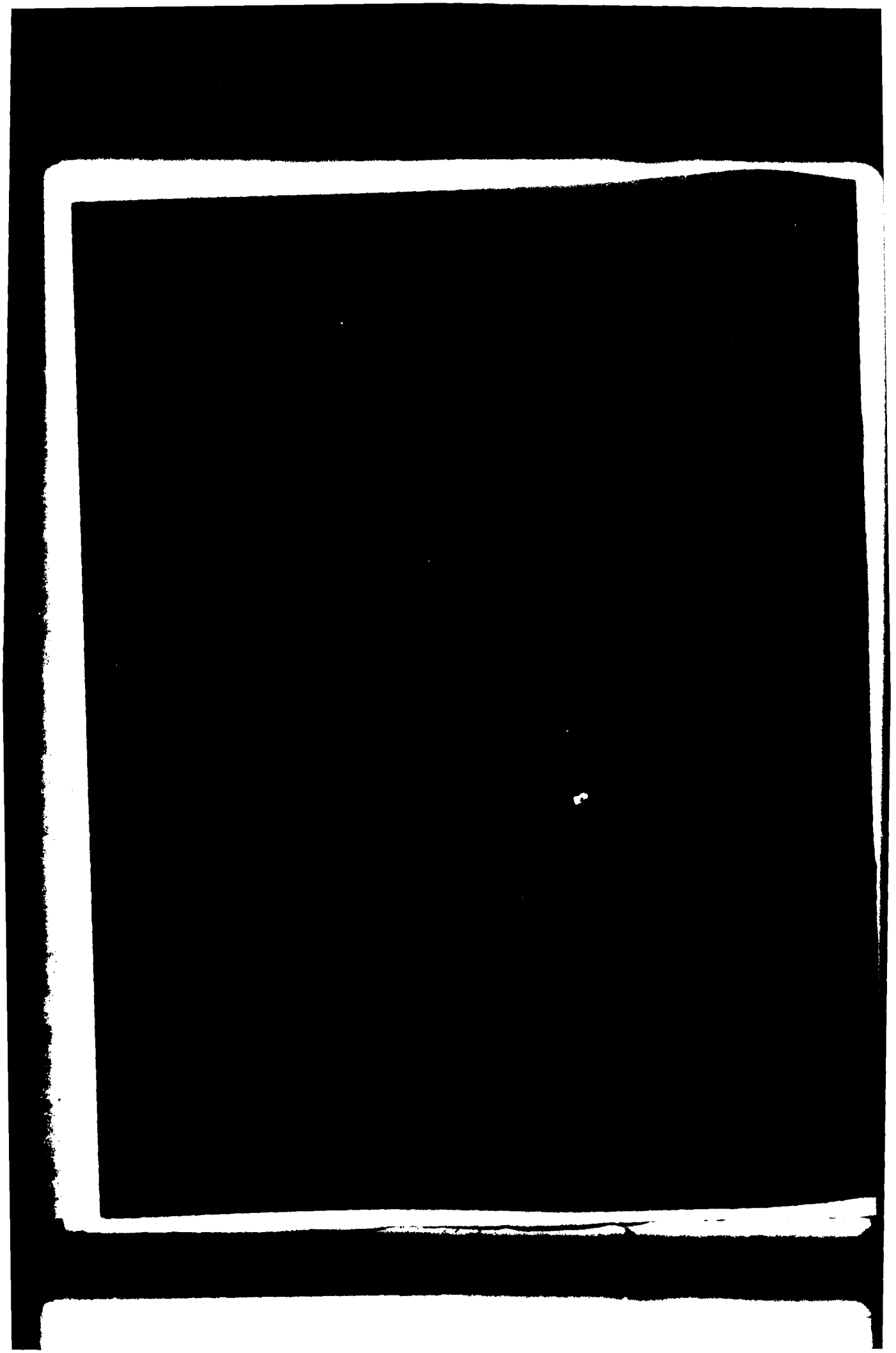
AD-A112 718 MASSACHUSETTS INST OF TECH LEXINGTON LINCOLN LAB F/G 17/2
NETWORK SPEECH SYSTEMS TECHNOLOGY PROGRAM.(U)
SEP 81 C J WEINSTEIN F19628-80-C-0002
UNCLASSIFIED ESD-TR-81-336 NL



END
DATE FILMED
4 82
DTIC





MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LINCOLN LABORATORY

NETWORK SPEECH SYSTEMS TECHNOLOGY PROGRAM

ANNUAL REPORT
TO THE
DEFENSE COMMUNICATIONS AGENCY

1 OCTOBER 1980 — 30 SEPTEMBER 1981

ISSUED 4 FEBRUARY 1982

LEXINGTON MASSACHUSETTS

ABSTRACT

This report documents work performed during FY 1981 on the DCA-sponsored Network Speech Systems Technology Program. The two areas of work reported are: (1) communication system studies in support of the evolving Defense Switched Network (DSN) and (2) design and implementation of satellite/terrestrial interfaces for the Experimental Integrated Switched Network (EISN). The system studies focus on the development and evaluation of economical and endurable network routing procedures. Satellite/terrestrial interface development includes circuit-switched and packet-switched connections to the experimental wideband satellite network. Efforts in planning and coordination of EISN experiments are reported in detail in a separate EISN Experiment Plan.

CONTENTS

NETWORK SPEECH SYSTEMS TECHNOLOGY

I. INTRODUCTION AND SUMMARY

This report documents work performed during FY 1981 on the DCA-sponsored Network Speech Systems Technology Program. The two major areas of work reported are: (1) communication system studies in support of the evolving Defense Switched Network (DSN) and (2) design and implementation of satellite/terrestrial interfaces for the Experimental Integrated Switched Network (EISN). Efforts in planning and coordination of EISN experiments are reported in detail in a separate EISN Experiment Plan.

The FY 81 system study effort, described in Section II, was directed at the development and evaluation of routing procedures with the potential to satisfy the dual DSN requirements of economy under normal operating conditions and endurability under stress. A generic class of networks called mixed-media networks was defined to include the basic DSN characteristics of a large number of small switches, a mix of satellite and terrestrial transmission media, and exchange of routing control information over a common channel signaling (CCS) network. Three classes of routing procedures – referred to as (1) mixed-media routing, (2) adaptive-mixed-media routing, and (3) precedence flooding – were developed to satisfy the requirements of economy and endurability for mixed-media networks. These procedures are all designed to make effective use of a mix of terrestrial transmission media and demand-assigned satellite media. We have made qualitative comparisons among these three classes of routing procedures, and have developed computer-based network analysis tools for quantitative evaluation of the performance of these routing procedures in multi-node networks. These tools include large-scale network analysis programs, as well as an algorithm which was developed as an extension of FY 80 work on sequential routing[1] for calculation of point-to-point blocking probability. Preliminary results have been obtained for test mixed-media networks, indicating that adaptation of routing tables after certain types of network damage provides improvements in worst case node-to-node blocking probabilities.

Section III describes FY 81 efforts in design and implementation of satellite/terrestrial interfaces for EISN. These include a packet/circuit interface (PCI) between a circuit-switched telephone network and the wideband satellite network (WB SATNET), and an Internet Packet Gateway (IPG) to connect the Experimental Data Network at Defense Communications Engineering Center (DCEC) to the WB SATNET. The functional designs for these systems were developed in FY 80. The FY 81 effort comprised detailed design and development of the hardware and software. The largest effort was in the development of the PCI (formerly referred to as Applique C), since the IPG (formerly called Applique D) is a relatively simple extension of similar gateway developed under DARPA sponsorship. To support satellite/terrestrial integration and routing experiments using the PCI, a Telephone Office Emulator (TOE) has been developed to produce voice and signal traffic in T1 carrier format. After initial experimentation, a commercial digital circuit switch will replace the TOE as input to the PCI. Prototype units of the PCI and TOE are currently operational and being used for EISN experiments at Lincoln. A second set of units is under construction, so that matching experimental facilities at the Defense Communications Engineering Center (DCEC) and Lincoln will be provided for satellite/terrestrial communication experiments over the WB SATNET.

As part of the FY 81 program, Lincoln has carried on a limited involvement with voice conferencing as a continuation of previous efforts.[2] This activity has been limited to the continuation for the first six months of FY 81 of our subcontract with Bolt Beranek and Newman, Inc. (BBN) to provide human-factors consultation in support of the WWMCCS Secure Voice/Graphics Conferencing (SVGC) Test and Evaluation Program. The Naval Ocean System Center (NOSC) is the lead organization in this program. It was anticipated that testing would have started at NOSC in early FY 81, and the BBN would contribute to the design and administration of the tests and in the evaluation of early test results. As it turned out, testing at NOSC did not begin during the contract period, and the BBN consulting effort consisted of the preparation and delivery to NOSC of an informal document discussing statistical properties of system ratings obtained during Phase II of the earlier study, and of various

statistical analysis techniques that were applied. In addition, the summary proposes a statistical test that was not used in the earlier evaluation, and which appears to have particular relevance to the treatment of data of the type generated in the rating experiments to be undertaken at NOSC.

II. COMMUNICATION SYSTEM STUDIES FOR THE DSN

A. INTRODUCTION AND SUMMARY

Communication system studies during FY 81 have been directed at the development of routing procedures potentially applicable to the DSN. The goal is to develop a routing procedure which provides the economical performance under normal operating conditions and the endurability desired in the DSN. This procedure should employ distributed control, use both demand-assignment multiple-access (DAMA) satellite and land links effectively, use common channel signaling and switch CPU processing bandwidth effectively, and sustain network operation with intentional or unintentional signaling errors. In FY 81, we first identified those generic characteristics which are unique to DSN-like networks and defined a new class of networks called mixed-media networks which have these generic characteristics. We then[3] carried out a comprehensive review of previous routing procedures and investigated the applicability of these procedures to the problem of routing in mixed-media networks. Particular attention was focused on the POLYGRID routing scheme used in the current AUTOVON. Although many previous routing procedures contain elements which are applicable to mixed-media networks, new procedures are needed to specifically address the dual DSN requirements of endurability and low cost. Three classes of routing procedures developed to address these requirements are described here. The major distinguishing feature of these procedures is that satellite and land links are treated separately and uniquely both when routing tables are created and when calls are actually routed.

The three routing procedures developed for mixed-media networks are referred to as: (1) mixed-media routing, (2) adaptive-mixed-media routing, and (3) precedence flooding. Mixed-media routing uses fixed routing tables. Survivability under stress is obtained by completely avoiding or by routing calls away from busy or destroyed Earth stations and by allowing high-priority calls to try a large number of routes. Call-processing rules used with mixed-media routing employ spill-forward control, a new type of call processing called remote Earth station querying, or crankback. Information needed for

call routing is assumed to be transmitted between nodes in a call request message with a variable-length header. The sequential routing algorithm described in the previous Annual is a special case of mixed-media routing, where crankback is permitted. Adaptive-mixed-media routing is identical to mixed-media procedures except routing tables are automatically updated or adapted whenever the network is damaged. Precedence flooding routes high-priority calls using flooding and low-priority calls using mixed-media routing. Flooding has been used successfully in tactical networks and does not require routing tables. We have made qualitative comparisons among these three classes of routing procedures and have developed tools for quantitative comparisons. These tools include steady-state analysis models, call-by-call simulations, and an algorithm for calculating point-to-point blocking probabilities from link blocking probabilities. This algorithm was developed through an extension of analytic work on sequential routing reported last year. We have also carried out a computer analysis of some small test networks, obtained some preliminary quantitative results, and defined a plan for further investigation.

In the following sections we first review important DSN characteristics and define mixed-media networks. Next we review problems with the application of current AUTOVON POLYGRID routing to the DSN and describe the three new classes of routing procedures. We then review our plan for evaluating these procedures, describe tools we have developed and are developing, and present some preliminary results.

B. MIXED-MEDIA NETWORKS

In order to provide a basis for comparison and evaluation of candidate DSN routing procedures, we have defined a generic type of network referred to as a mixed-media network. The DCA has developed four alternative approaches[4] as candidate methods for satisfying the DSN objectives of (1) survivable and edurable telecommunications for high-priority DoD users and (2) economical routine service to lower priority users during non-stress conditions. The characteristics of mixed-media networks are defined to include the most important features relevant to routing that are common to the four alternate DSN approaches. These features include use of a large number of small

switches, a mix of media, and CCS interswitch signaling. Therefore we have defined mixed-media networks to include many small switches which exchange routing information over a common channel signaling network. They also include both terrestrial connectivity and satellite connectivity obtained using many small earth stations located near switches. These earth stations provide access to one or more broadcast satellites. Channel capacity in these satellites may be assigned using DAMA techniques or using fixed-capacity assignments.

An example of a mixed-media network which includes 16 nodes and 2 satellites is illustrated in Fig. II-1. Dots in this figure represent nodes without earth stations, and solid triangles represent nodes with earth stations. Implicit in this figure is a common channel signaling (CCS) network which parallels the network of voice trunks and which may be obtained using voice trunks

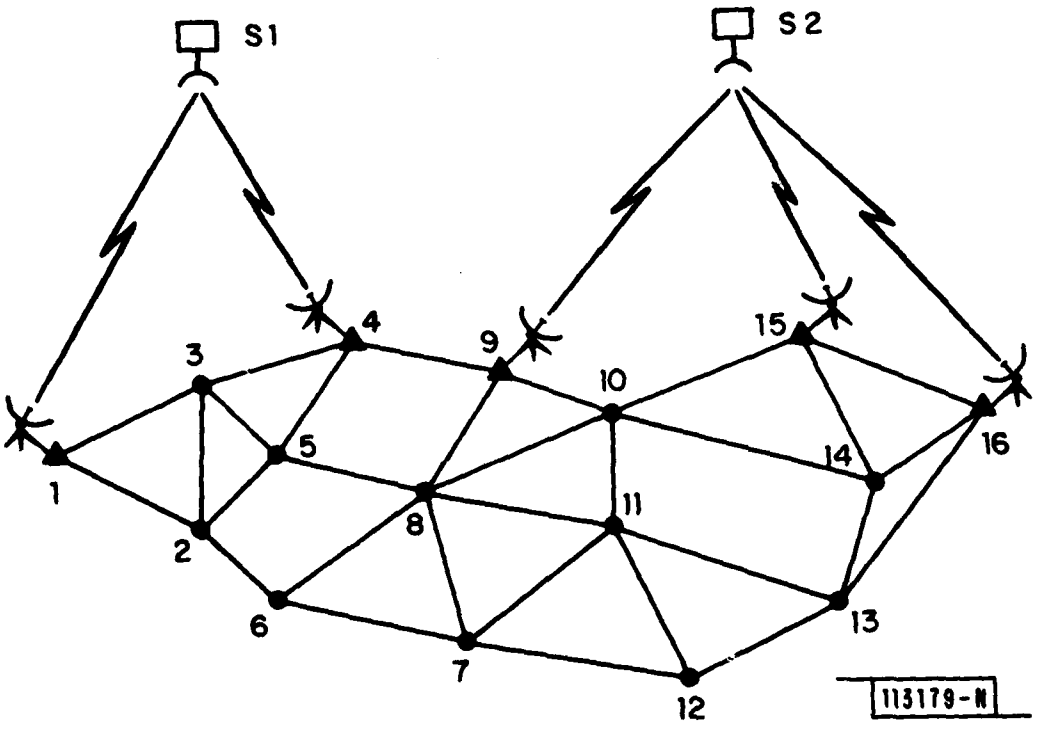


Fig. II-1. A mixed-media network with 2 satellites, 5 earth stations, and 16 nodes.

and part of the satellite bandwidth. Two nodes in this mixed-media network (nodes 1 and 4) include earth stations which can access satellite S1, and three nodes (nodes 9, 15, and 16) include earth stations which can access satellite S2. DAMA access to satellite S2 allows the number of calls routed over S2 between nodes 9 and 15, nodes 9 and 16, and nodes 15 and 16 to vary dynamically depending on the offered traffic until the total bandwidth required for all calls exceeds the channel capacity of satellite S2. At that point, new calls could be blocked or routed via land links, or existing calls could be preempted by higher priority calls. This type of fully variable DAMA access is not always possible because earth stations typically have bandwidths which are less than the bandwidth of the satellite transponder. In many situations it is thus necessary to limit both the number of calls routed through each earth station and the total number of calls routed through each satellite. Access to S2 via fixed-capacity assignments involves assigning a fixed proportion of the satellite bandwidth representing a fixed number of voice trunks separately to link nodes 9 and 15, nodes 9 and 16, and nodes 15 and 16. These satellite links are then used as "wires in the sky" with fixed capacities.

C. PROBLEMS WITH POLYGRID ROUTING

The transition from AUTOVON to the DSN would be simplified if POLYGRID routing as used in AUTOVON could be used without change. Unfortunately, POLYGRID routing has a number of important deficiencies with respect to application in mixed-media networks. First, it is heavily tailored to the POLYGRID structure of CONUS AUTOVON which is expensive to replicate in the DSN. The use of relatively simple routing controls (route control digits), the concept of a home grid in which routing is restricted, and the methods used to generate routing tables[5] are all predicated on the existence of a POLYGRID network with extensive short- and long-haul connectivity. This network is expensive because it includes not only the basic POLYGRID pattern but also a network of long-distance trunks which is overlaid on top of this pattern.[6]

A second major problem with POLYGRID routing is that the simple method used to limit path lengths and prevent loops (passing route control digits between switches) makes it very difficult to accommodate satellites. There are currently no DAMA access satellites in AUTOVON. Route control digits can

be used to prevent more than one or two lateral or backward movements in AUTOVON because a strict definition of forward and backward can be made. Such a definition is not simple to make in mixed-media networks which include DAMA satellites. In such networks, all earth stations should be considered to be the same distance apart in a routing procedure designed to make the best use of network resources because a short satellite hop uses no more network resources than a long satellite hop. Calls should thus be permitted to be routed "backwards" toward an earth station in a geographic direction which is away from the destination switch. In addition, calls should be routed away from blocked earth stations over the shortest path to the destination and calls should possibly be prevented from being routed towards blocked earth stations. This is neither possible nor necessary with POLYGRID routing. POLYGRID routing also does not guarantee that land routes are available when all long-distance trunks are destroyed because this is highly unlikely in AUTOVON where long-distance trunks are terrestrial. In a mixed-media network it is possible that all long-distance satellite links could be destroyed at once. It is essential that alternate land routes are available when this occurs. POLYGRID routing also includes no mechanism for preventing a voice call from traversing more than one or two satellite hops. Such a mechanism is needed in mixed-media networks to limit the total delay on voice calls.

A practical problem with POLYGRID routing is that the current procedure for developing routing tables is time consuming and unsuited for use in iterative-design algorithms or near-real-time updating of routing tables. Current programs used to generate POLYGRID routing tables require at least an hour of central processor time on a large computer. An ability to update DSN routing tables quickly and accurately would simplify network design and analysis and also would simplify the mechanics of adding and deleting switches and/or links. In addition it would make near-real-time routing table updating feasible.

D. ROUTING PROCEDURES FOR MIXED-MEDIA NETWORKS

1. Mixed-Media Routing

The simplest routing procedures which we are developing and evaluating and which are designed with the goal of satisfying DSN requirements are called

mixed-media procedures. These procedures use fixed routing tables and call-processing rules which employ spill-forward control, a new type of control called remote earth station querying, and crankback. Their major distinguishing characteristic is that satellite and land links are treated separately and uniquely both when routing tables are created and when calls are actually routed through a network. For example, call-processing rules use information about the status of key nodes with satellite earth stations to route calls. In addition, routing tables indicate whether the shortest path to the destination switch via a given outgoing link includes a satellite hop. The use of fixed routing tables and local signaling in these procedures enhances routing security and automatically protects unaffected parts of a network when intentional or unintentional signaling errors occur in specific locations. It also minimizes signaling and switch CPU processing bandwidth requirements and minimizes switch and signaling hardware requirements in general. These simplifications may, however, result in the necessity of larger and more trunk groups than are required by procedures which automatically adapt routing tables or by procedures which use flooding techniques to route calls.

Routing tables and call-processing rules used in mixed-media routing procedures are designed to:

(a) Provide alternate land and satellite routes for survivability.

(b) Maintain network operation when all satellite links are destroyed.

(c) Allow alternate earth stations to be used when the most desirable earth station is not available.

(d) Limit the total number of satellite hops allowed in a call path.

(e) Prevent loops and shuttles in the call path.

(f) Prevent excessively long routes.

(g) Allow more routing flexibility for high-priority calls than for routine calls.

These design goals are self explanatory except possibly for item (c). As noted previously, full DAMA satellite access allows all earth stations to use the satellite channel whenever this channel is not filled to capacity. Unfortunately, full DAMA access is not always practical because the bandwidth of individual earth stations is typically less than that of the satellite transponder. Calls which cannot be transmitted over the satellite because a desired receiving earth station is busy must thus be allowed to use an alternate receiving earth station, and calls which cannot be transmitted because a desired transmitting earth station is busy must be allowed to use an alternate transmitting earth station.

a. Routing-Table Generation

Routing tables are used in each node of a network to determine the next link in the call path to a given destination. Routing tables for DSN-size mixed-media networks can be produced automatically by computer using a three-stage process. Figure II-2 and Tables II-1, -2, and -3 illustrate how this process produces a routing table for use in node 10 of the network shown in Fig. II-2 when the destination is node 16. The numbers in parentheses in Fig. II-2 are link lengths in miles.

In the first stage of the routing-table generation process, the shortest path to the destination via each outgoing link of the originating node is found using a shortest path algorithm. This path can include both land and satellite links. The outgoing links are then ordered on the basis of path lengths and placed in a list. The list for the network of Fig. II-2 is presented in Table II-1. This table does not contain a complete description of each shortest path. It only contains the first links in each path and some additional information on path length and on the first earth station and satellite in each path. For example, it indicates that the first link on the shortest path to node 16 (10-9-S2-16) is link 10-9, and that this path uses two links, is 316 miles long, uses satellite S2, and uses the earth station at node 9. In this and in following tables, first links are ordered primarily by the total number of links in the associated shorted path. A further ordering by total path length is made if link counts are equal. This ordering metric attempts to use the least number of switches and links

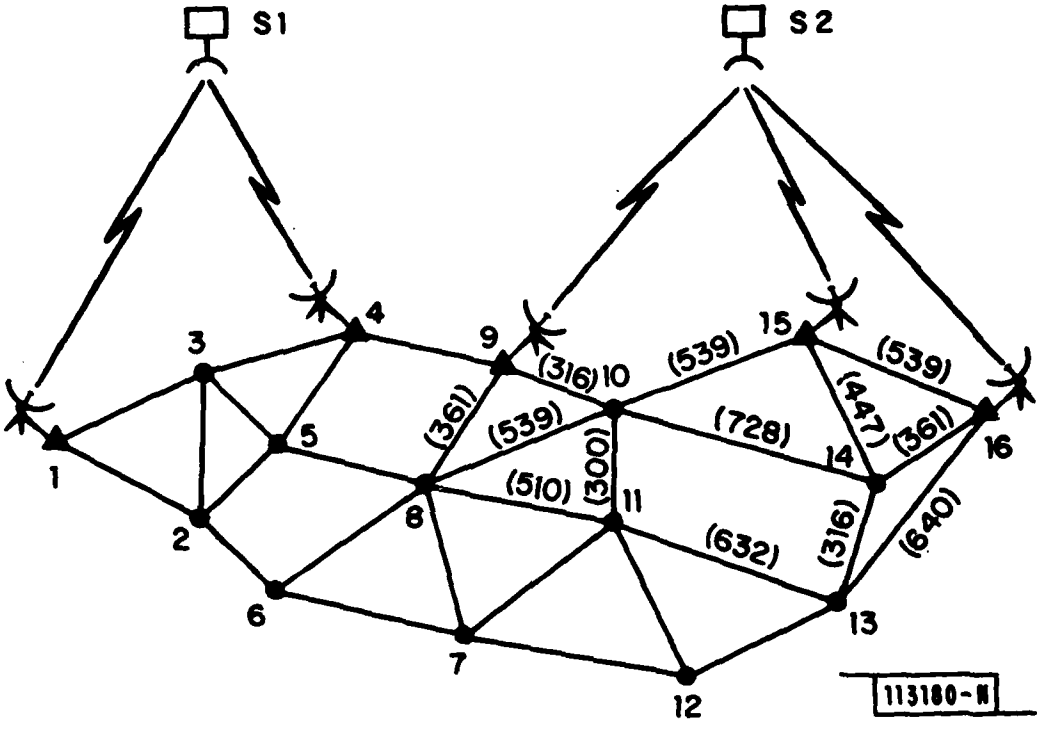


Fig. II-2. A mixed-media network with the lengths of links used to route calls from node 10 to node 16 indicated in parentheses.

TABLE II-1

OUTGOING LINKS FROM NODE 10 OF FIG. II-2 ORDERED BY THE NUMBER OF LINKS IN AND TOTAL LENGTH OF THE SHORTEST PATH TO NODE 16

Link	Number of Links in Path	Total Length of Path (mi)	Use Satellite	Earth Station
10-9	2	316	yes (S2)	9
10-15	2	539	yes (S2)	15
10-14	2	1089	no	–
10-8	3	900	yes (S2)	9
10-11	3	1572	no	–

TABLE II-2

OUTGOING LINKS FROM NODE 10 OF FIG. II-2 ORDERED BY THE NUMBER OF LINKS IN AND TOTAL LENGTH OF THE SHORTEST LAND PATH TO NODE 16

Link	Number of Links in Path	Total Length of Path (mi)
10-15	2	1078
10-14	2	1089
10-11	3	1572
10-8	4	2321
10-9	5	2459

TABLE II-3

FINAL ROUTING TABLE USED IN NODE 10 OF FIG. II-2 FOR DESTINATION NODE 16

Link	Use Satellite	Earth Station
10-9	yes (S2)	9
10-15	yes (S2)	15
10-15	no	–
10-14	no	–
10-8	yes (S2)	9
10-11	no	–
10-8	no	–

Shortest Land Route = 2 Links

Shortest Satellite Route = 2 Links, 1 Hop

to complete each call. The distance of a satellite hop used to create Table II-1 was zero and a satellite hop was counted as one link. In an operational network, the distance of a satellite hop would be set high enough to prevent calls between nearby users from being routed over a satellite but low enough to load the satellite to capacity. Table II-1 could be used as a routing table except it only applies to a network in which all satellites and earth stations are operating and not destroyed or blocked.

In the second stage of the routing-table generation process a list is produced as in the first stage, but all satellites are removed from the network and only land paths are considered. The list produced for the network in Fig. II-2 is presented in Table II-2. Note that the first link on the shortest path to node 16 (10-15-16) is now link 10-15. Table II-2 could be used as a routing table except it applies only when satellite S2 is blocked or destroyed.

In the third stage of the routing-table generation process, the lists produced in stages one and two are combined to form a final routing table. The routing table for the network in Fig. II-2 is presented in Table II-3. All first links on shortest land and satellite paths are treated separately in this table. This provides alternate land routes when satellites or earth stations are destroyed. For example, link 10-8 appears in both the fifth and seventh positions of this table. The lower numbered position corresponds to a satellite path and the higher to a land path. The maximum number of entries allowed in this routing table was eight and four of these eight had to be associated with land routes. These numbers were chosen to prevent excessively long routes but to provide adequate alternate routing if satellite S2 is destroyed. Only seven entries are included because some of the entries in Tables II-1 and -2 are duplicates and use identical paths and because the first link on the longest land path (entry five in Table II-2) was excluded to prevent excessively long routes. Such routes must be prevented because they can greatly degrade network performance, especially when the offered traffic level is high. The number of links in the shortest land path and the number of links and hops in the shortest satellite path are included in the routing table for use by the call-processing rule. This supplementary information is used to limit the length of alternate paths and also to prevent excessive delay.

The above routing-table generation process is characterized by the maximum number of routing-table entries, by the number of entries associated with shortest land paths, by the maximum allowable path length, and by the metric used to order paths. These characteristics depend on network loading and topology and must be determined empirically. The metric used above is simple and similar to the metric used in AUTOVON and in a number of packet-switched networks. Other metrics which use information on sizes, loading, and survivability of links and switches are also being examined.

b. Call-Processing Rules

(1) Overview

The three types of call-processing rules used in mixed-media routing procedures are (a) spill forward, (b) remote earth station querying, and

(c) single-stage crankback. Spill-forward control is simple and used in many networks including CONUS AUTOVON. Each switch either blocks a call or routes it to another switch not yet in the call path. Remote earth station querying is a new type of call processing made possible by common-channel signaling. A switch sends a query message to an earth station in the shortest path to the destination to determine the status of the earth station and any required satellite. If the earth station and satellite are available, the call is routed normally. If not, the call is routed on the first link of the shortest path to the destination which does not require the earth station or satellite. Remote earth station querying thus prevents calls from being routed toward blocked or destroyed earth stations or satellites without adapting or updating routing tables. It does this without increasing signaling or switch CPU processing bandwidth excessively. Single-stage crankback allows a call request blocked at a node to be cranked back to the previous switch in the call path. Many more alternate paths are examined with single-stage crankback than with spill-forward control. If a path exists to the destination, it is thus more likely to be found with crankback than with spill-forward control.

All three types of call-processing rules require a common-channel signaling network in which switches automatically sense failure or destruction of attached links and adjacent switches and in which earth stations sense the failure of associated satellites. Call request messages are sent over this network to establish call paths. These messages contain a special header in addition to the called number and call priority. The variable-length component of this header includes a trace which is a list of switches and satellites currently in the call path. It also includes a list of known blocked or destroyed earth stations and satellites. Each switch in the call path adds data to this part of the header. The fixed-length component of the header contains the maximum number of links allowed in land and satellite routes to the destination and the maximum number of satellite hops allowed. Only the originating switch places data in this part of the header. The limits on path lengths and number of satellite hops are calculated using both the supplementary data stored in routing tables and detour lengths which may vary for calls with different priorities. Information in the call request header is used by switches to

prevent loops and shuttles (a loop between two nodes), to route calls away from or avoid blocked or destroyed earth stations and satellites, to limit the length of alternate routes, to prevent excessive delay, and to direct the cranking back of calls.

(2) Spill-Forward Control

Each switch either blocks a call or routes it to another switch not yet in the call path when spill-forward control is used. The next switch in the path is chosen by sequentially examining the links listed in the routing table and using the first free link which does not lead to a node already in the call path. (A list of nodes in the call path is available in the call request header.) In mixed-media routing, each switch also examines the list of blocked or destroyed earth stations and satellites in the call request header and skips over routing-table entries with associated shortest paths that include these earth stations and satellites. Calls are thus routed away from unavailable earth stations and satellites. Each switch also compares the number of links and satellite hops in the call path to the maximum number stored in the header to determine whether another link or satellite hop is allowed. If no more links are allowed, the call is blocked. If no more satellite hops are allowed, routing-table entries associated with shortest satellite paths are skipped.

Call paths established using spill-forward control are illustrated in Figs. II-3 and -4. The mixed-media network in these and the following figures includes 14 switches, 2 earth stations, and 1 satellite. The call path in Fig. II-3 illustrated by a dotted line is the path established when no links are busy. Starting with switch 1, each switch places itself on the trace in the header and routes the call using the first routing-table entry. When the call request arrives at node 14, the trace includes 1-2-3-S1-11 and the list of blocked earth stations and satellites is empty. The first switch in the call path (switch 1 in Fig. II-3) computes the maximum number of links allowed in land routes (5 + detour of 2 = 7), the number of links allowed in satellite routes (4 + detour of 2 = 6), and the number of satellite hops allowed (1). These limits are determined using the supplementary data in the routing table and detour

ROUTING TABLE IN NODE 1
FOR DESTINATION NODE 14

113181-N

LINK	USE SATELLITE ?	EARTH STATION
1-2	YES (S1)	3
1-4	YES (S1)	3
1-7	NO	–
1-4	NO	–
SHORTEST LAND ROUTE = 5 LINKS		
SHORTEST SATELLITE ROUTE = 4 LINKS, 1 HOP		

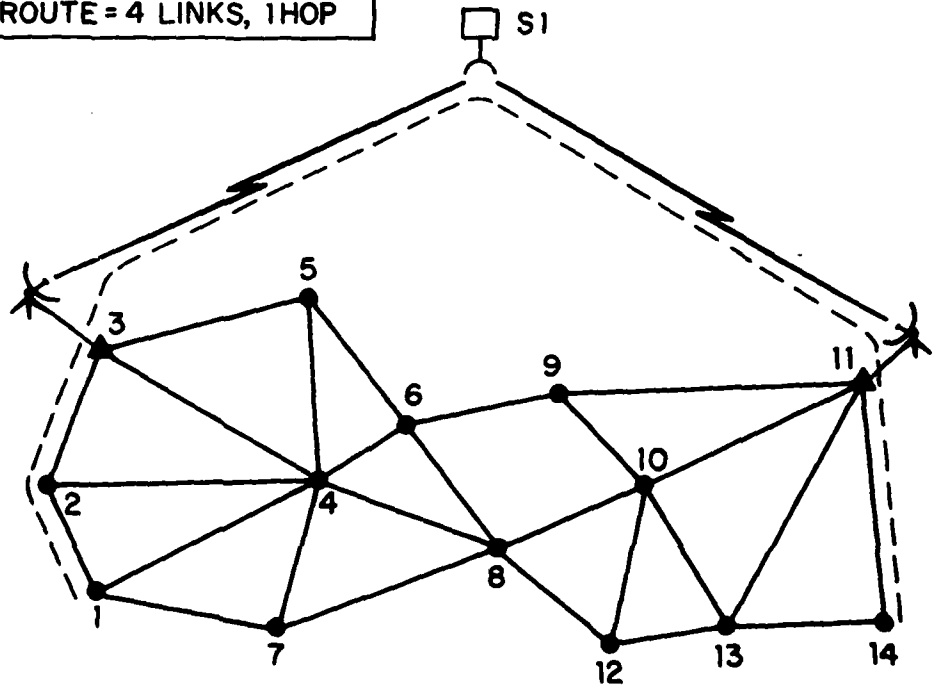


Fig. II-3. Call path from node 1 to node 14 when the earth stations and the satellite are available and mixed-media routing is used.

113162-N

ROUTING TABLE IN NODE 3
FOR DESTINATION NODE 14

LINK	USE SATELLITE?	EARTH STATION
3-11	YES (S1)	3
3-4	NO	-
3-5	NO	-

ROUTING TABLE IN NODE 4
FOR DESTINATION NODE 14

LINK	USE SATELLITE?	EARTH STATION
4-3	YES (S1)	3
4-5	YES (S1)	3
4-2	YES (S1)	3
4-8	NO	-

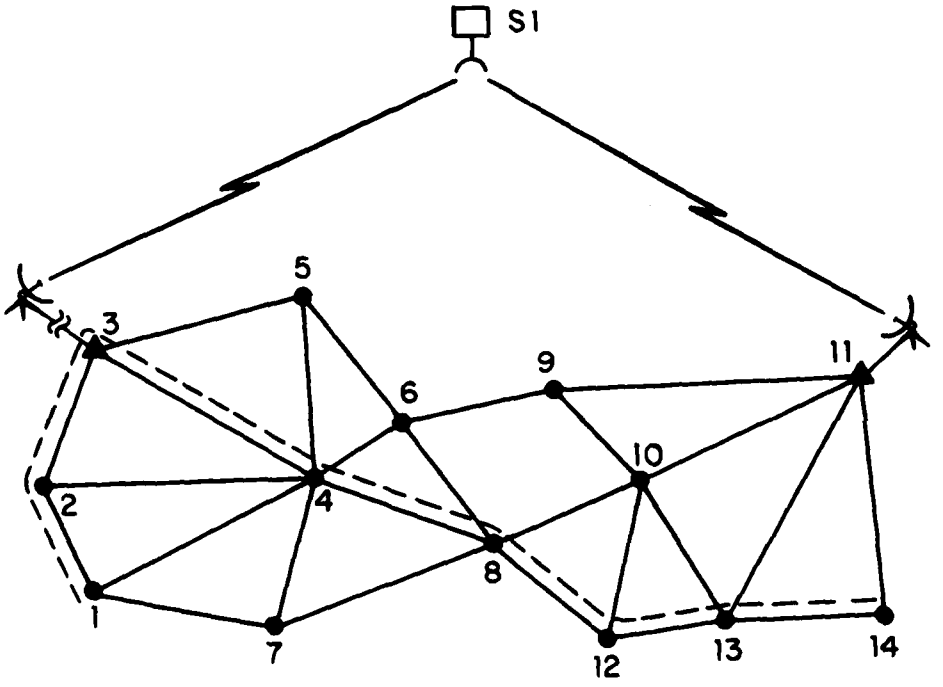


Fig. II-4. Call path from node 1 to node 14 when spill-forward control and mixed-media routing are used and the earth station at node 3 is blocked.

lengths. These lengths may be longer for high-priority calls. In addition, more routing-table entries may be used for high-priority calls.

The call path in Fig. II-4 was established when the earth station at node 3 was busy. The call request message travels to switch 3 as in Fig. II-3. Here link 3-11, the first routing-table entry in node 3, is blocked. Switch 3 recognizes this, places the earth station in node 3 onto the list of blocked earth stations in the call request header, and routes the call to node 4 using the second routing-table entry, 3-4. The switch at node 4 skips the first three routing-table entries because they lead to shortest paths which include the blocked earth station at node 3 (this earth station is on the list in the call request header). The switch at node 4 routes the call to node 8 using the last routing-table entry, 4-8. All other nodes route the call using the first entry in their routing table for destination node 14.

When the call request arrives at node 14, the trace includes 1-2-3-4-8-12-13 and the list of blocked earth stations includes the earth station at node 3. Note that if the list of blocked earth stations had not been passed to switch 4 the call would have been routed from switch 4 to switch 5 on link 4-5, the first routing-table entry which does not lead to a node in the trace. The call would then have been routed to nodes 6, 8, and 12. At node 12 it would have been blocked and lost because the call path would have been equal to the maximum allowable length stored in the call request header (seven links).

In the above example, including a list of blocked earth stations and satellites in the header prevented the call from being lost. In general, including this list reduces path lengths by preventing calls from wandering around a network to reach blocked or destroyed earth stations. In the above example, the call had to detour to node 3 to find out that the earth station at node 3 was busy. This detour can be prevented with remote earth station querying.

(3) Remote Earth Station Querying

Remote earth station querying is similar to spill-forward control except an earth station on the shortest path to the destination can be remotely queried using common-channel signaling. If the earth station or the satellite required

is blocked or destroyed, the call is not routed to the earth station. If earth station and satellite are available, the call is routed normally.

Figure II-5 illustrates a call path when remote earth station querying is used and the earth station at node 3 is blocked. The detour evident in Fig. II-4 has been completely eliminated, and the call is routed directly to node 14. In Fig. II-5, the first node in the call path (node 1) sends a query message to the earth station at node 3 because this earth station is on the shortest path to the destination (see first routing-table entry). The return message from switch 3 indicates that the earth station is blocked. Switch 1 thus adds the earth station to the list of blocked earth stations in the call request header and routes the call on link 1-7, the first routing-table entry not associated with a path that includes the earth station at node 3. The other switches also route calls using routing-table entries not associated with paths which include the earth station at node 3. When the call request arrives at node 14, the trace includes 1-7-8-12-13 and the list of blocked earth stations includes the earth station at node 3.

(4) Single-Stage Crankback

Single-stage crankback is similar to spill-forward control except a call request blocked at a node is routed backwards to the previously visited node. This type of crankback searches many more alternate paths than spill-forward control. It is identical to sequential routing[1,7] except the controls described above have been added to prevent loops and shuttles, limit path lengths, and prevent routing to blocked earth stations. These controls are needed to prevent excessively long alternate routes which could degrade network performance. In addition, the routing-table generation process has been defined.

Figure II-6 illustrates the call path when the earth station at node 3 is blocked, links 3-4 and 3-5 are blocked, and single-stage crankback is used. The call request message travels first to node 3. Here no outgoing links are available and the call would be blocked if spill-forward control were used. Single-stage crankback, however, allows the call to crankback to node 2. Before it is cranked back, switch 3 adds the earth station at node 3 to the list of

ROUTING TABLE IN NODE 1
FOR DESTINATION NODE 14

113183-N

LINK	USE SATELLITE?	EARTH STATION
1-2	YES (S1)	3
1-4	YES (S1)	3
1-7	NO	-
1-4	NO	-

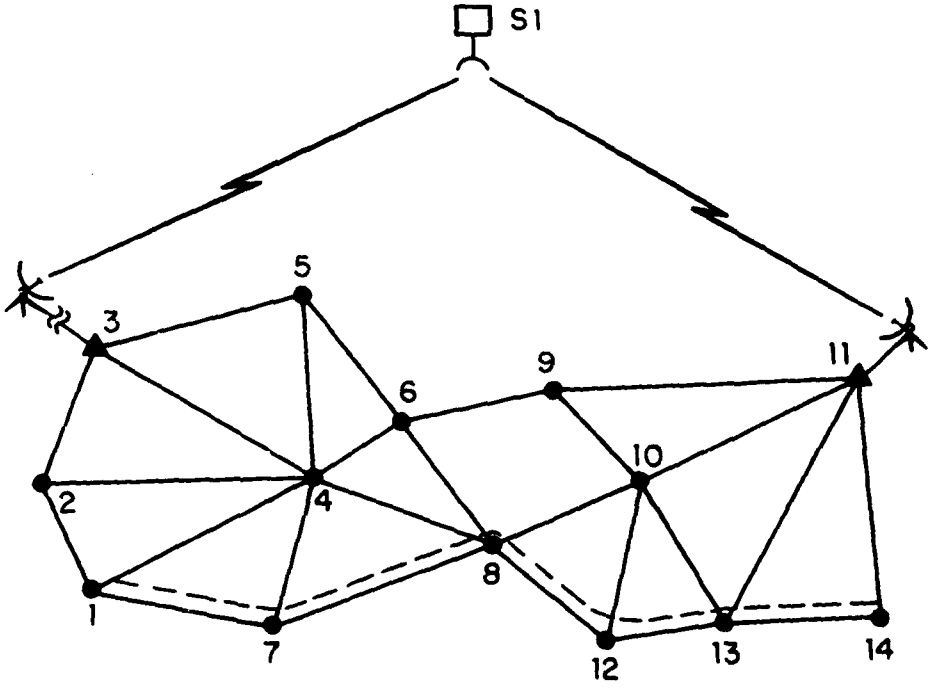


Fig. II-5. Call path from node 1 to node 14 when remote earth station querying and mixed-media routing are used and the earth station at node 3 is blocked.

ROUTING TABLE IN NODE 2
FOR DESTINATION NODE 14

113184-N

LINK	USE SATELLITE?	EARTH STATION
2-3	YES (S1)	3
2-4	YES (S1)	3
2-4	NO	-
2-1	NO	-

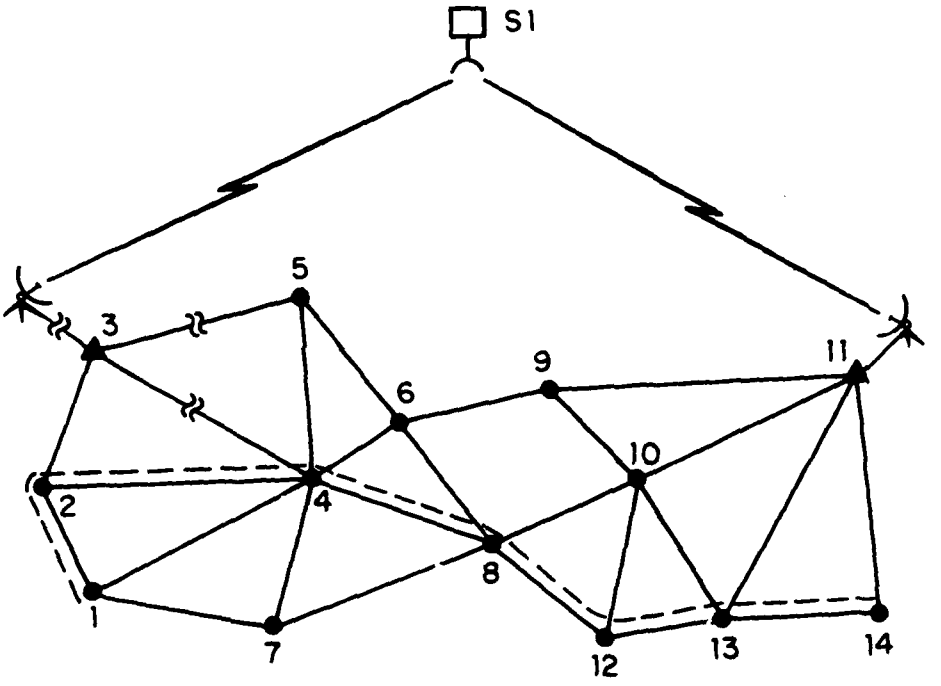


Fig. II-6. Call path from node 1 to node 14 when single-stage crankback control and mixed-media routing are used, the earth station at node 3 is blocked, and the links between nodes 3 and 4 and between nodes 3 and 5 are blocked.

blocked earth stations in the call request header. When the call request arrives back at node 2, it thus indicates that the earth station at node 3 is busy. This prevents the call from being routed toward node 3 and causes the call to be routed over land links to node 14. The trace when the call arrives at node 14 includes 1-2-3-2-4-8-12-13 and the list of blocked earth stations includes the earth station at node 3. Note that the call pathe would have been identical to that in Fig. II-4 if link 3-4 had been free.

(5) Special Features

All call-processing rules used in mixed-media routing procedures allow MLPP (multi-level precedence and preemption), least cost routing through other networks, and trunk group reservation for high-priority traffic.

MLPP involves assigning each call a priority and giving preferential treatment to higher priority calls. For example, a high-priority call can preempt the trunk used by a lower priority call if all the outgoing links listed in the routing table are busy and a lower priority call is currently using a trunk on one of these links (friendly preemption). Alternatively, it can preempt a trunk used by a lower priority call on any link whenever all trunks on that link are busy (ruthless preemption). In either case, the high-priority call uses the preempted trunk to reach the next switch where it may again preempt a lower priority call to obtain another trunk. The completion of one high-priority call may thus lead to the termination of connections between many lower priority callers. In a more complex form of preemption, the high-priority call can preempt the complete call path of a lower priority call which is already in progress and which terminates at the destination of the high-priority call. This results in the termination of only one ongoing conversation to complete a high-priority call.

Another feature of MLPP is that high-priority calls are allowed to use all entries in mixed-media routing tables, while low-priority calls are only allowed to use the first LP entries (the number LP depends on network topology and must be determined empirically). High-priority calls are also allowed to traverse longer paths before being blocked. For example, if the shortest land path to a destination is two links long, then high-priority calls may be allowed

to use paths which include as many as five links while low-priority calls may only be allowed to use paths which include three or fewer links.

Least cost routing allows a call to be completed using an alternate network when it cannot be completed over the most desirable network. For example, an originating switch may route a call over the Federal Telecommunication system (FTS) or over the Bell System Direct Long Distance Dialing (DDD) network when it is blocked in the DSN. In addition, calls which are preempted may be rerouted using off-net facilities and the order in which alternate networks are tried may vary with time of day to take advantage of time-of-day rate changes.

Trunk group reservation for high-priority traffic is designed to reduce the incidence of preemption in networks which include MLPP. The final few trunks in each trunk group are reserved for high-priority calls. Trunk group reservation has been used previously[8] to reserve trunks for direct-routed traffic at the expense of alternate-routed traffic and found to be beneficial. Reservation of trunks for high-priority traffic has not, however, been tested using the mix of traffic expected in the DSN.

2. Adaptive-Mixed-Media Routing

Adaptive-mixed-media routing procedures are identical to the previously described static-mixed-media procedures under normal operating conditions. When the network is damaged, however, routing tables are automatically updated to enhance survivability.

The procedures used to update routing tables in adaptive-mixed-media routing are similar to those currently used in the ARPANET.[9] Each node stores global information which describes the network topology. This information is updated only when the network topology changes (switches or links are added, removed, or destroyed). Update information is transmitted from nodes which detect a change using flooding on a secure common-channel signaling network. Routing tables are recomputed in each node when an update is received using a shortest path algorithm and procedures similar to those used to generate the original set of routing tables. This procedure is relatively simple, but it has many advantages over other adaptive procedures. It provides adaptive,

distributed routing based on global information, but does not suffer from any inherent adaptation rate, stability, or convergence problems. Such problems could occur in procedures such as the failsafe protocol[10] or the old ARPANET routing procedure[11] which adapt continuously on the basis of network performance. Other advantages of this procedure are that routing tables for users with different priorities are easy to maintain, a table of unreachable nodes is automatically provided, and the two main components of the procedure have been tested in operational networks. The flooding scheme used to transmit database updates has been used since 1979 in the ARPANET (the required signaling bandwidth in the ARPANET is less than 250 bps). The procedure used to calculate new routing tables is similar to that used in AUTOVON, in the network analysis programs developed at DCEC,[12] and in the ARPANET. Conservative assumptions concerning switch processing capabilities lead to times to compute new routing tables of from half a second to one minute.

Another type of adaptive-mixed-media routing procedure updates routing tables when changing traffic patterns overload part of a network. This is more complex than adapting only when the network is damaged. Network performance information must be obtained and transferred between nodes, the metric used to generate routing tables must be sensitive to trunk group loading and size, and the possibility exists for instability or convergence problems. This type of procedure will only be examined if the other simpler routing procedures prove to be inadequate.

The main disadvantage of adaptive-mixed-media routing is the extra complexity in switches it requires and the necessity of transmitting global information describing topological changes around the network. This information must be transmitted and received accurately to sustain network performance. Although it can be heavily encrypted and protected and switches can employ defensive programming to protect themselves against intentional or accidental signaling errors, it would be difficult to completely eliminate the possibility that such errors could significantly disrupt network operation. Evidence that this would occur infrequently is available in the relatively good performance of the ARPANET. The possibility of network disruption caused by signaling errors and the increased complexity and cost of this adaptive procedure must

be weighed against its improved capability to reconstitute the network. The major goal of our evaluation of this procedure is thus to determine the extent of this improvement.

3. Precedence Flooding

A routing procedure which employs saturation routing or flooding to route high-priority traffic and mixed-media routing to route low-priority traffic is also being evaluated. The flooding scheme described in Reference 14 is used except controls are added to limit path lengths. When the destination switch is more than one link away, the originating switch sends a SEARCH message for the destination to all adjacent switches over the common-channel signaling network. Each switch which receives a SEARCH message stores it in a memory and forwards it to all adjacent switches except the one which sent the SEARCH message. Each switch forwards only the first SEARCH message it receives, and SEARCH messages propagate through the network. Some die out at the edges of the network, but if a path exists to the destination, at least one reaches the destination. At the destination, the path for routing the call is selected using information added to the SEARCH messages as they passed through the network. This information includes the number of links in each path, the number of satellite hops, and, possibly, a measure of link loading in the path. The switch at the destination then sends a return message back to the originating switch over the selected path. The return message establishes the call path.

This routing procedure is attractive because the shortest path is actually found by an exhaustive search. Its intuitive appeal is also supported by the good performance of flooding in Reference 14. Flooding provided the best grade of service when compared to the routing procedures currently used in CONUS and European AUTOVON and to an adaptive procedure which used backward learning based on link count. Flooding performed better than these procedures under normal traffic loads, overloads, and when the simulated network was damaged. It also adapted quickly and automatically after damage. Flooding is reasonable in the DSN only for high-priority traffic because of its large signaling and switch CPU processing requirements. Conservative assumptions

made in Reference 13 indicate that 2400-baud common-channel signaling links (obtained, for example, from single voice trunks) and PDP-11/34-type switch controllers could support at most only the projected high-priority AUTOVON traffic for 1985 (roughly 1300 Erlangs). The use of 4800-baud common-channel signaling links would increase the allowable traffic which can be routed using flooding to 2500 Erlangs.

E. EVALUATION STRATEGY

1. Overview

Work so far has focused on defining the major characteristics of candidate routing procedures for the DSN and on developing tools and techniques for evaluating these procedures. The routing procedures being evaluated are:

(a) Mixed-Media Routing

  (1) Spill-Forward Control

  (2) Remote Earth Station Querying

  (3) Single-Stage Crankback

  (4) Single-Stage Crankback and/or Remote Earth Station Querying for High-Priority Calls and Spill-Forward Control for Low-Priority Calls

(b) Adaptive-Mixed-Media Routing

(c) Flooding for High-Priority Traffic and Mixed-Media Routing for Low-Priority Traffic.

Tools needed to examine the performance of these procedures include both analytic models and call-by-call simulations. Mixed-media networks which are small enough to make analysis tractable have already been created. The offered traffic in these networks has been set to that predicted for the DSN, and trunk groups have been fixed for the desired grade of service with this offered traffic when a reference routing procedure (e.g., mixed-media routing with spill-forward call processing) is used. The other routing procedures can

be compared using these networks on the basis of the following important network characteristics:

(1) Node-to-Node Grade of Service Distribution (by priority)

(2) Incidence of Preemption (by priority)

(3) Signaling Traffic

(4) Call Completion Time (by priority)

(5) Router/Controller Processing and Memory Requirements

(6) Dynamic Behavior After Network Damage and Overload

(7) Vulnerability to Accidental and Intentional Signaling Errors

(8) Cost

The first six characteristics must be evaluated under normal network operating conditions, with various types of traffic overloads, and after and during various types and amount of network damage.

The steady-state performance of the simplest types of mixed-media and adaptive-mixed-media routing can be determined using a steady-state analysis program developed by DCEC[12] but modified for these procedures. The characteristics of these procedures which lead to best performance are being chosen using this modified program. These characteristics include the metric used to calculate path lengths when generating routing tables, the maximum number of links allowed in the detours which are part of alternate routes for high- and low-priority calls, and the number of routing-table entries for land and satellite routes for high- and low-priority calls. Once these characteristics have been selected, the performance of the various types of mixed-media routing can be evaluated. The benefits of DAMA satellite access can also be examined.

A call-by-call simulation program must be used to examine the dynamic behavior of all routing procedures, to validate the results obtained with the steady-state analysis program, and to examine the incidence of preemption. We have obtained the FORTRAN simulator used in Reference 14. We are

modifying this simulator to include mixed-media and adaptive-mixed-media routing. It already includes flooding routing and MLPP with five priority levels. It also allows the network to be dynamically overloaded and damaged while grade of service, signaling traffic, and incidence of preemption are monitored. Comparisons performed with this simulator are essential to the formulation of recommendations concerning routing procedures for DSN.

2. Development of Analysis Tools

We have written programs in RATFOR to generate routing tables used in mixed-media and adaptive-mixed-media routing procedures. Routing tables for all nodes in a 16-node network can be generated in less than 2 s on an Amdahl 470 computer. These tables allow a maximum of 10 alternate links out of each node. We have modified the DCEC analysis program to use these tables and spill-forward call processing. Programs have also been written to generate mixed-media test networks, and test networks have been generated using topological and offered traffic data for one tentative DSN configuration which was provided by DCEC. This tentative configuration is one of many being explored. One test network includes the 20 nodes from the 100-node DCEC configuration which originate and terminate the greatest amount of traffic. These 20 nodes account for roughly 60 percent of the traffic in the DCEC configuration, and 5 of these nodes include earth stations for one satellite. We have also examined the GPSS FORTRAN event-by-event simulator and the Computer Sciences Corporation AUTOVON event-by-event simulator and decided that these simulators are not adequate for this study. We have instead obtained the call-by-call simulator used in Reference 14.

We have begun to examine the effect of updating routing tables in small networks after the networks suffer various types of damage. The steady-state grade of service between all nodes in the networks has been estimated using the steady-state network analysis program provided by DCEC. Preliminary data indicate that the average node-to-node grade of service may not improve after routing tables are updated, but that the maximum node-to-node grade of service does improve. For example, in one test network with 15 nodes, 1 satellite, and 2 earth stations, destroying 1 node increased the average grade of

service from 0.1 to 0.18. The maximum node-to-node grade of service increased from 0.28 to 1.0, and a number of node pairs could no longer communicate. After routing tables were updated, the average grade of service was unchanged but the maximum node-to-node grade of service decreased from 1.0 to 0.55 and all node pairs could again communicate. These results were obtained using a network with very little connectivity and a restrictive routing procedure (forward routing[15]). Further work needs to be done using mixed-media networks and mixed-media routing procedures before any general conclusions are made.

3. Further Work on the Sequential Routing Algorithm

The Sequential Routing Algorithm was described in last year's Annual Report[1] and in Reference 7. This algorithm includes a call-processing rule called single-stage crankback which is part of one of the candidate routing procedures being explored for the DSN. Switches contain routing tables and are directed, when necessary, to exhaustively search all permissible routes to find and exploit any remaining connectivity in a damaged network. It is very difficult to evaluate the performance of this algorithm analytically because of the number of alternate routes in all but the simplest networks.

Analytic results for a class of highly symmetric tree networks were presented last year. Equations were developed which allow the point-to-point grade of service between nodes in a symmetric tree network to be determined when either sequential routing or spill-forward routing is used and the link blocking probabilities are known in each link and fixed. A pair of new recursive program designs have been developed to extend this analysis to more conventional networks which use either sequential routing or spill-forward call-processing rules. These recursive algorithms are generically related to other recent work on recursive formulas.[16,17,18] There are, however, certain important differences. The other recent methods require all permissible routes between nodes to be precomputed either by hand or algorithmically and then apply combinatorial calculations to determine the point-to-point blocking probability. The new algorithm requires only routing tables. It automatically cycles through all permissible routes (including all levels of crankback, in the

case of the sequential routing algorithm) and maintains a running subtotal of the contribution of each route to the net blocking probability, without actually requiring explicit display or manipulation of the details of routes. The recursive code which performs these calculations is extremely compact and is described below. The new algorithm can be used as a network design tool to compute point-to-point steady-state blocking probabilities in networks with trunk groups which will be sized for a known blocking probability. It can also be used to estimate the point-to-point blocking probability of a small amount of high-priority traffic which is routed through a network that is operating with a steady traffic load and that has known trunk group blocking probabilities.

The input to the recursive code is a set of routing tables for the network and the blocking probability for each link in the tables. The routing tables are defined as a general four-dimensional array rttbl [N] [N_D] [N_{link}] [3]. Figure II-7 illustrates the rectangular array formed by the rightmost two dimensions of rttbl for a particular choice of source node N_S and destination node N_D. Notice that the entries in the array each contain only one of the links leaving N_S and not a complete route from N_S to N_D. The links are rank ordered in the array, with the uppermost one being the most desirable first link for routes to this particular N_D. The quantity p(blk) is the individual blockage probability for the associated link, and the flag field will be used in a future dynamic simulation of the sequential routing algorithm. The number of rows in the array is great enough to accommodate all the links associated with the

109374-N

BEST LINK	p(blk)	FLAG
SECOND LINK	p(blk)	FLAG
THIRD LINK	p(blk)	FLAG

Fig. II-7. Routing table for a particular source-destination pair.

most heavily interconnected node in the network; for all other nodes one or more of these rows are filled with zeroes.

The collection of routing tables covering all the possible destination nodes from the particular source node N_S consists of a three-dimensional stack of permuted versions of Fig. II-7, with certain links added or deleted as appropriate. This three-dimensional array is the full set of routing tables that would be stored in the memory of the routing processor at node N_S in a real network. In the present analytical approach, however, the functions of the individual nodal processors are to be accomplished within one computer by successive calls to the same program, using a different three-dimensional set of routing tables for each call. This is mechanized by means of the fourth or outermost dimension of rttbl, which indexes each possible source node N_S in the network under consideration.

The routing table for a particular (N_S-N_D) pair can contain zero or more destination links which connect N_S and N_D directly, as well as zero or more intermediate links which end at one of the intermediate nodes N_{11}, N_{12}, as shown in Fig. II-8. We assume that the numbering of the link blocking probabilities corresponds to their positions in the (N_S-N_D) routing table, so that $p_1 < p_2 < \ldots < \ldots < p_m$; the link ending at N_{11} is preferred over that ending at N_{12}; and so on.

The model used in this analysis for spill-forward routing requires the following assumptions for a particular (N_S-N_D) call attempt:

(a) A link leaving N_S is tried if and only if all links above it in the routing table are blocked;

(b) When an intermediate node is reached, control passes ("spills forward") to it as if it had become the source node; and

(c) When a call request is blocked at all exits from a node, it is dropped and reinitiated by the originator (i.e., crankback is not allowed).

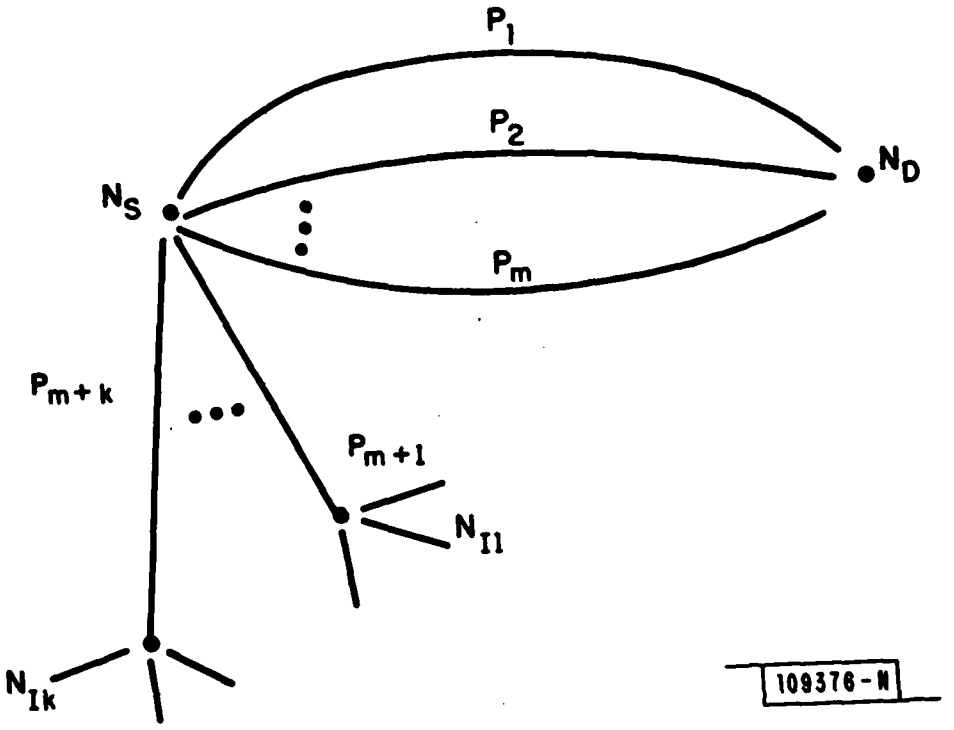


Fig. II-8. Node and link types.

Under these assumptions, the node and link model illustrated in Fig. II-8 leads to the following equations:

$$P(S,D) = \text{Prob}\{a(N_S\text{-}N_D) \text{ call attempt is blocked}\}$$

$$= 1 - \text{Prob}\{\text{the call is completed on one of the } \textit{permissible routes}\}$$

$$= 1 - (1 - p_1) - p_1(1 - p_2) -$$

$$\vdots$$

$$-p_1p_2\cdots p_{m-1}(1 - p_m)$$

$$-p_1p_2\cdots p_m(1 - p_{m+1})\,[1 - P(I1, D)]$$

$$\vdots$$

$$-p_1p_2\cdots p_{m+k-1}(1 - p_{m+k})\,[1 - P(Ik, D)] \quad . \qquad (1)$$

Note that Eq. (1) explicitly uses recursion, in that P( ) is computed for each intermediate node exactly as if that intermediate node were itself the source node for a call request to N_D. Expanding and cancelling terms in Eq. (1), one obtains the compact recursive function illustrated in Fig. II-9.

The model used in this analysis for the sequential routing algorithm requires the following assumptions for a particular (N_S-N_D) call attempt:

(a) A link leaving N_S is tried if and only if all links above it in the routing table are blocked,

(b) A call request which is blocked at all exits from an intermediate node is cranked back to the closest preceding node having any still-untried links, and

(c) A call request is ultimately blocked if and only if every possible (N_S-N_D) route is blocked.

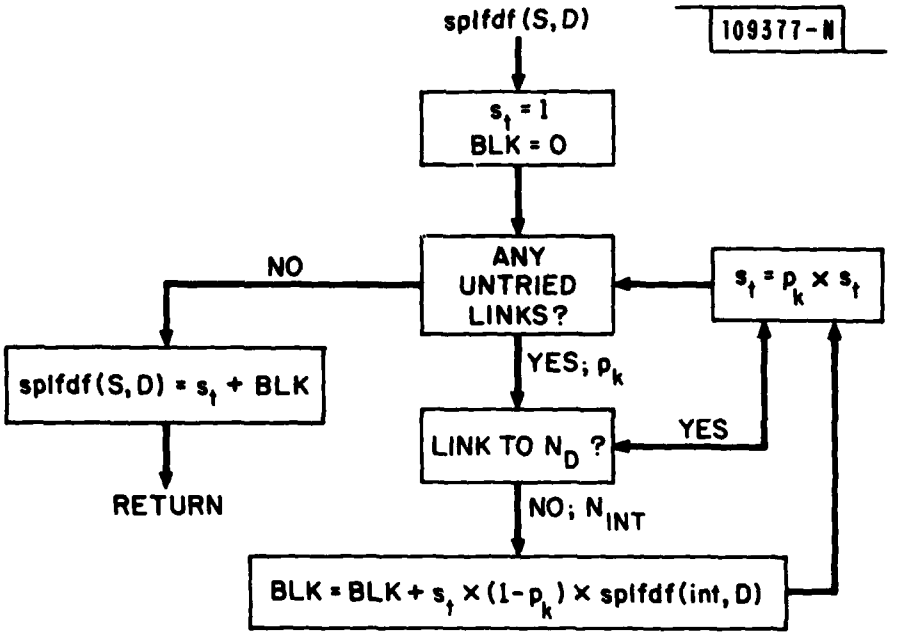


Fig. II-9. Flow chart for blocking with spill-forward routing.

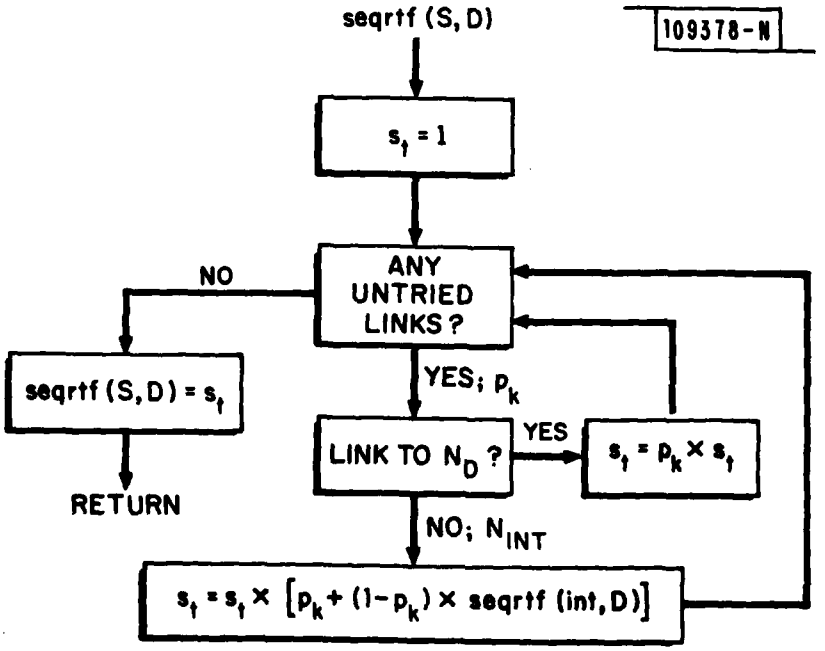


Fig. II-10. Flow chart for blocking with sequential routing.

These assumptions lead to the following equations and thence to Fig. II-10:

$$P(S, D) = \text{Prob}\{\text{all possible S-D routes are blocked}\}$$

$$= p_1 p_2 \cdots p_m$$

$$\times [p_{m+1} + (1 - p_{m+1})\, P(I1, D)]$$

$$\cdot$$

$$\cdot$$

$$\cdot$$

$$\times [p_{m+k} + (1 - p_{m+k})\, P(Ik, D)] \quad .$$

These recursive blocking analyses can be used to compare the end-to-end blocking performance of the sequential routing algorithm with that of standard spill-forward routing in the same network for the same set of routing tables and link blocking probabilities. The procedure to be followed is to use a main program which calls splfdf (S, D) and seqrtf (S, D) and computes their quotient. An initial version of this program has been written, and work is currently in progress on the mechanization of methods for conveniently varying routing tables to permit rapid calculation of performance comparisons over network conditions (as reflected by the link blocking probabilities) ranging from lightly loaded to heavily stressed.

III. SATELLITE/TERRESTRIAL INTERFACE DEVELOPMENT

A. INTRODUCTION

The DSN is expected to include satellite connectivity overlaid on a network of digital circuit switches located on or near military bases. This satellite overlay will be designed to provide reduced cost for long-distance communication in non-stress conditions, as well as a degree of flexibility and endurability under limited-stress conditions. A primary EISN objective is experimental demonstration of the feasibility of this satellite overlay concept through representative experiments in a network much smaller than the DSN itself. During FY 80, Lincoln Laboratory developed functional designs[1] for satellite/terrestrial interfaces and switch simulators to support these scaled DSN experiments. Detailed design and implementation of the hardware and software for these units has been carried out during FY 81.

The scaled DSN experiments are being carried out on an evolving system which will increase in size and capability as new sites and facilities are added. Figure III-1 shows an architectural diagram of this experimental scaled DSN, which sets a system context for description of the FY 81 Lincoln development efforts. The system derives its satellite connectivity through the wideband satellite network (WB SATNET), a demand-assigned packet satellite network which supports both the scaled DSN experiments and DARPA-sponsored experiments in multi-user packet speech communication. Local scaled DSN equipment is being installed first at DCEC and at Lincoln, and later at the U.S. Army Communications Command facilities at Ft. Huachuca, Arizona, and Ft. Monmouth, New Jersey, and at the U.S. Air Force Rome Air Development Center (RADC).

Two major phases of the development of the experimental system are indicated. The first major phase of scaled DSN development involves the DCEC and Lincoln sites, equipped as shown within the upper dashed contour of Fig. III-1. Substantial FY 81 Lincoln efforts have been devoted to development of the packet/circuit interface (PCI) and telephone office emulator (TOE). The PCI serves as a satellite/terrestrial interface between a digital circuit switch

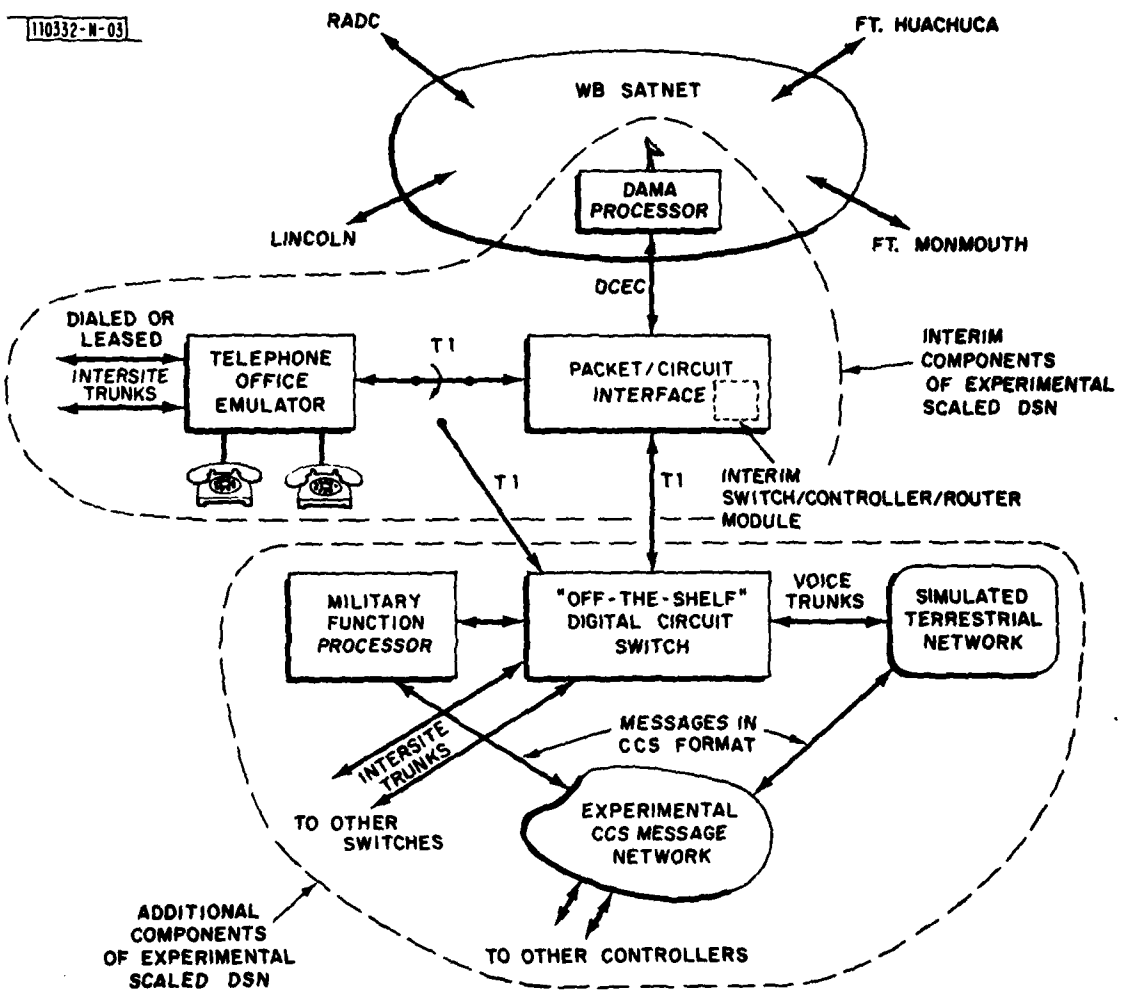


Fig. III-1. Scaled DSN site equipment.

typical of the DSN and the packet-switched WB SATNET. The PCI converts voice circuits embedded in a T1 carrier to packet satellite format. This voice-circuit-based interface has previously been referred to as Interface Applique C (for circuits); the new terminology is more descriptive and provides a distinction between interface functions, and routing and control functions to be discussed below.

The PCI is designed to be connected to an "off-the-shelf" digital circuit switch, as shown in Fig. III-1. The TOE has been developed to act as an experimental test vehicle to provide T1 format traffic to the PCI prior to the availability of a real digital circuit switch. The TOE will accommodate local telephones as well as dialed-up connections serving as intersite terrestrial trunks between (for example) DCEC and Lincoln. The PCI and TOE jointly provide a rudimentary switching capability which can support alternate satellite/terrestrial routing experiments among the sites. The PCI includes a Digital Equipment Corporation PDP-11/44 computer which has sufficient excess processing capacity to support an interim switch/controller/router module (see Fig. III-1) for these experiments. Specific experiment scenarios are detailed in the EISN experiment plan.

The DSN will eventually accommodate data as well as voice, and a set of experiments is to be directed at issues in internetwork packet data communications including terrestrial and satellite links. During FY 81, Lincoln has developed an internet packet gateway (IPG) to allow communication between the Experimental Data Network (EDN) at DCEC and the WB SATNET. The IPG (formerly referred to as Interface Applique D, for Data) shares PDP-11/44 hardware with the PCI, and forms a gateway for DoD standard Internet Protocol (IP) packets.

The remaining scaled DSN components shown in Fig. III-1 are to be the subject of design and development efforts beginning in FY 82. These include "off-the-shelf" commercial digital circuit switches typical of those to be used in the DSN, "applique" controller/routers to implement special military DSN functions (e.g., MLPP, mixed-media routing strategies, adaptive reconstitution of routing tables) which are not available in commercial switches, a simulated terrestrial network to create the effect of embedding the five-node scaled DSN

in a much larger net, and an experimental CCS message to allow communication of routing messages among the sites. When commercial switches become available, the TOE will be utilized as shown to provide voice access for experimental users of the system.

The next three subsections describe in more detail the major FY 81 Lincoln development efforts with respect to the PCI, the TOE, and the IPG. The FY 81 detailed design and implementation efforts are based on functional designs which were developed in FY 80 and described in the 30 September 1980 Annual Report. The PCI represented the major portion of the overall development effort, and separate descriptions are given of its hardware and software development. Prototype units of the PCI, TOE, and IPG have all been implemented and shown to perform their basic functions correctly. A second set of units is currently under construction so that the Lincoln and DCEC sites will be provided with compatible equipment by January 1982 and two-site scaled DSN experiments can commence.

An overview block diagram of the satellite/terrestrial interface subsystems is shown in Fig. III-2. The Packet/Circuit Interface (PCI) is comprised of: (1) a packet circuit translator (PCT) which converts between digital voice plus interswitch signals in T1 format, and voice and signals in packet format; (2) an internet gateway processor (PDP-11/44 computer) which handles the flow of voice and signal packets according to the stream (ST)[19] protocol and the internet datagram protocol (IP), respectively; and (3) an interface which carries out front-end packet processing on the WB SATNET side. The Internet Packet Gateway (IPG) shares the gateway processor and WB SATNET interface, but includes a separate packet interface to the EDN. The interface serving each net (circuit-switch, EDN, and WB SATNET) includes a microprocessor-based I/O system (the UMC-Z80). Each of the three interfaces has network-specific software in its Z80, and special network-specific interface hardware. The hardware and software for the PCT required substantial FY 81 efforts, which will be described below. Figure III-2 indicates a switched arrangement wherein either the PCI or the IPG, but not both, are operational at any given time, as will be the case for most of our initial experiments. However, both can be operated at once, with the PDP-11/44 then utilized as a shared resource for voice and data traffic.

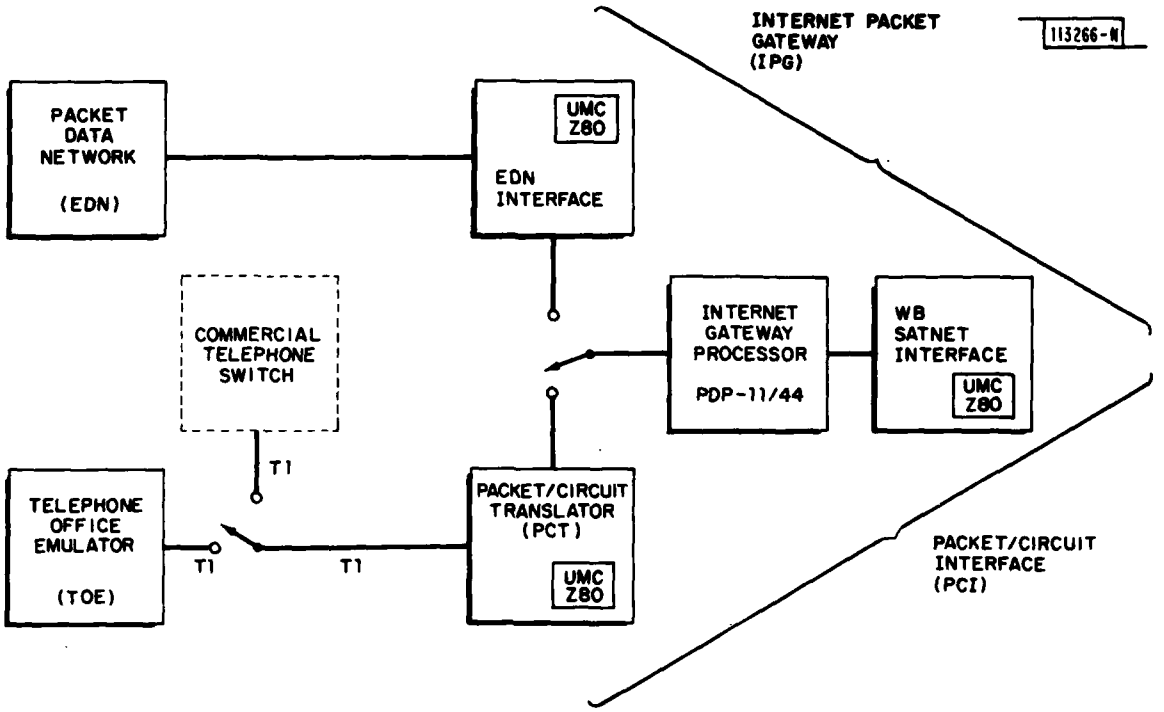


Fig. III-2. The components of the satellite/terrestrial interfaces.

B. PACKET/CIRCUIT INTERFACE

1. Overview and Hardware Description

The PCI is composed of a gateway, implemented in a PDP-11/44 computer, and the PCT. The PCT hardware, shown in Fig. III-3 is composed of a commercial digital channel bank (Northern Telecom DE-3, common equipment only), a commercial processor board (the UMC-Z80, with an associated memory board), and two processor boards designed and built at Lincoln. All these components can be mounted within the chassis of the gateway's PDP-11/44, but it has been decided to mount the channel bank, containing one of the Lincoln-made boards, in the same rack as the TOE, because the channel banks require a nonstandard rack width for flush mounting. The functions of the components will be traced starting at the left of Fig. III-3.

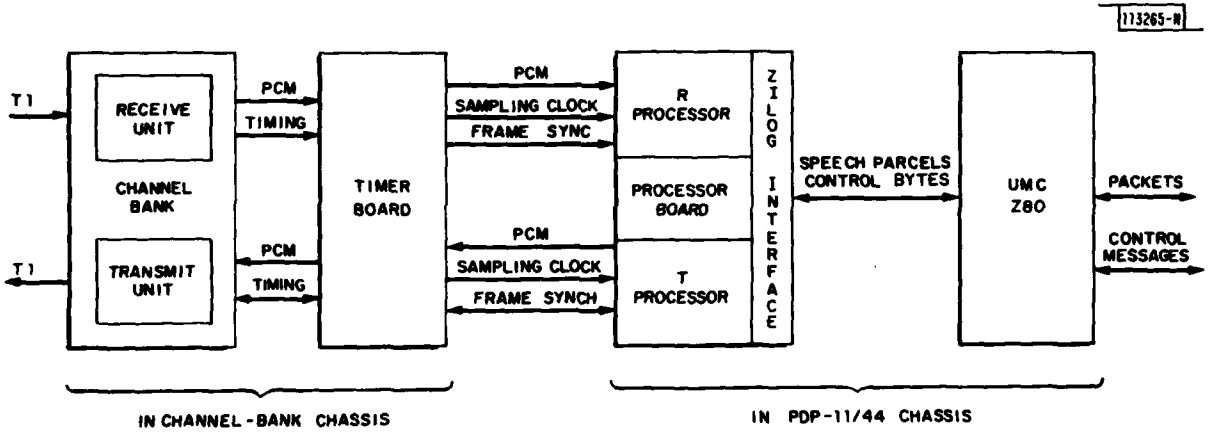


Fig. III-3. The PCT hardware components.

The T1 carriers are 1.544-Mbps digital telephone carriers, each capable of carrying 24 voice channels on a cable pair. The PCI, in its initial implementation, is designed to accept E&M signaling* for the voice circuits, and the channel for the E&M signals is derived by stealing the least significant bit of every twelfth 8-bit speech sample. Only a subset (i.e., 4 to 6) of the 24 voice channels will be serviced by the PCI/TOE combination. This will be enough channels to prove out the interface design and to conduct simple scaled DSN routing experiments.

The timer board is a Lincoln-built plug-in for the channel bank. It fits into a slot that would normally contain a channel unit. From the backplane of the channel bank it receives only power, but from the front panels of the channel bank's receive unit (RU) and transmit unit (TU) it receives all the timing signals generated in these units. The RU and TU had to be modified to extract these signals, but the modified units will still operate as ordinary RU and TU if no cable is attached to the connector on the front panel. The timer board has three tasks. The first is to pass the 1.544-Mbps multiplexed PCM stream from the channel bank to the processor board, or to pass PCM data from the processor board to the channel bank. The second is to provide clocks for the processor board that will enable it to pick out just the active channels from the PCM stream or to insert PCM samples into the active channels. Finally, the timer board transmits, and in some cases generates, synchronizing signals that keep the processor board and the channel bank synchronized as to the location of signaling frames.

The processor board is a Lincoln-built unit fabricated on a wirewrap board, sized to plug into the PDP-11/44 backplane. Its job is to do the per-sample processing of the speech, leaving the UMC-Z80 free to do the packet processing and signal translation. In the circuit-to-packet direction, the processor board partitions the samples into parcels of about 20 ms and detects whether a parcel contains speech or silence. It also locates the signaling bits (low-order bits of every twelfth sample) and determines whether the E&M

* The term E&M designates a particular interswitch signaling format used widely in the public telephone system.

signal is a 0 (on-hook) or a 1 (off-hook). It carries this out for every active channel in the T1 carrier. All the 8-bit samples from these channels are then forwarded to the UMC-Z80 in the order received. Following a parcel of such samples, the processor board appends one 8-bit control byte for each utilized channel. That byte contains coded information indicating the speech activity and the E&M signaling state. In the packet-to-circuit direction, the process is reversed. A parcel's worth of interleaved speech samples is preceded by one control byte for each utilized channel. The processor board examines the control byte and inserts the E&M signal value, 0 or 1, into every signal bit in the parcel's worth of samples sent to the channel bank. It also generates a parcel's worth of silence samples for each silent channel.

Communication between the processor board and the UMC-Z80 is carried out with DMAs. The DMAs have permuted address lines, enabling them to generate addresses in an order that will place the interleaved samples from the active channels into separate buffers, each containing the samples of one speaker in order. This relieves the UMC-Z80 of a very time-consuming addressing task. The basic tasks of the UMC-Z80 are those of signal translator and speech forwarder. These are described more fully in the section on software. In the circuit-switched network, connections are set up and taken down by E&M signals in the low-order bytes of speech samples. In the packet-switched satellite network, connections are set up and taken down by means of the interchange of protocol packets. The UMC-Z80 must translate between these two signaling protocols. In the circuit-switched network, once a connection is established, speech is transmitted by a synchronous stream of samples. In the packet-switched satellite network, speech is transmitted by a succession of packets, each containing one or more parcels' worth of speech and each preceded by a short header identifying the connection. The UMC-Z80 must append the headers in the circuit-to-packet direction and reconstitute the synchronous stream of samples in the packet-to-circuit direction. The latter operation involves buffering several packets to smooth out variations in transit time over the packet network.

The synchronizing schemes used between the channel bank, processor board, and UMC-Z80 have been designed so that the components can be

powered up in any order and automatically achieve synchronization. If synchronization is lost during operation, say, by an interruption of the T1 carrier, it will be automatically reestablished once the fault is corrected.

Figure III-4 is a block diagram of the processor board. On the left it communicates with the timer board and on the right with the UMC-Z80. In fact, it is really two independent processors that share only a common clock. The R processor receives data from the channel bank and prepares it for the UMC-Z80, while the T processor takes data from the UMC-Z80 and prepares it for transmission to the timer board. Between each of the processors and the timer board is a 16-deep FIFO, which serves as an elastic buffer between the synchronous channel bank and the asynchronous processor. The FIFOs allow the processors, including the UMC-Z80 to cease briefly to process individual speech samples and execute background functions, such as restarting the DMAs, without losing any of the speech samples. When four channels of speech are being processed at 8000 samples per second, the FIFOs provide 0.5 ms of breathing space for processor functions. At the clock speed used for the 8085 processors on the board, this time allows about 300 bytes of code to be executed. The synchronizing signals between the processor board and the channel bank (via the timer board) are also buffered to make them arrive in the proper time relation to the speech bytes.

Communication with the UMC-Z80 is accomplished by three Zilog chips that reside on the Z80 bus, an extension of the main processor bus on the UMC-Z80 board. One DMA device controls transfers in each direction through the dual-channel PIO device. The DMAs are shown with permuted address buses to perform the demultiplexing function mentioned earlier. The DMAs are programmed to transfer 724 bytes at a time (one control byte and one 180-byte speech parcel for each of four speakers). Because of the permuted wiring scheme, the number of voice channels presented to the UMC-Z80 must be a power of two. Changing the number of voice channels requires manual rewiring of the DMAs, but that can be accomplished in a few minutes.

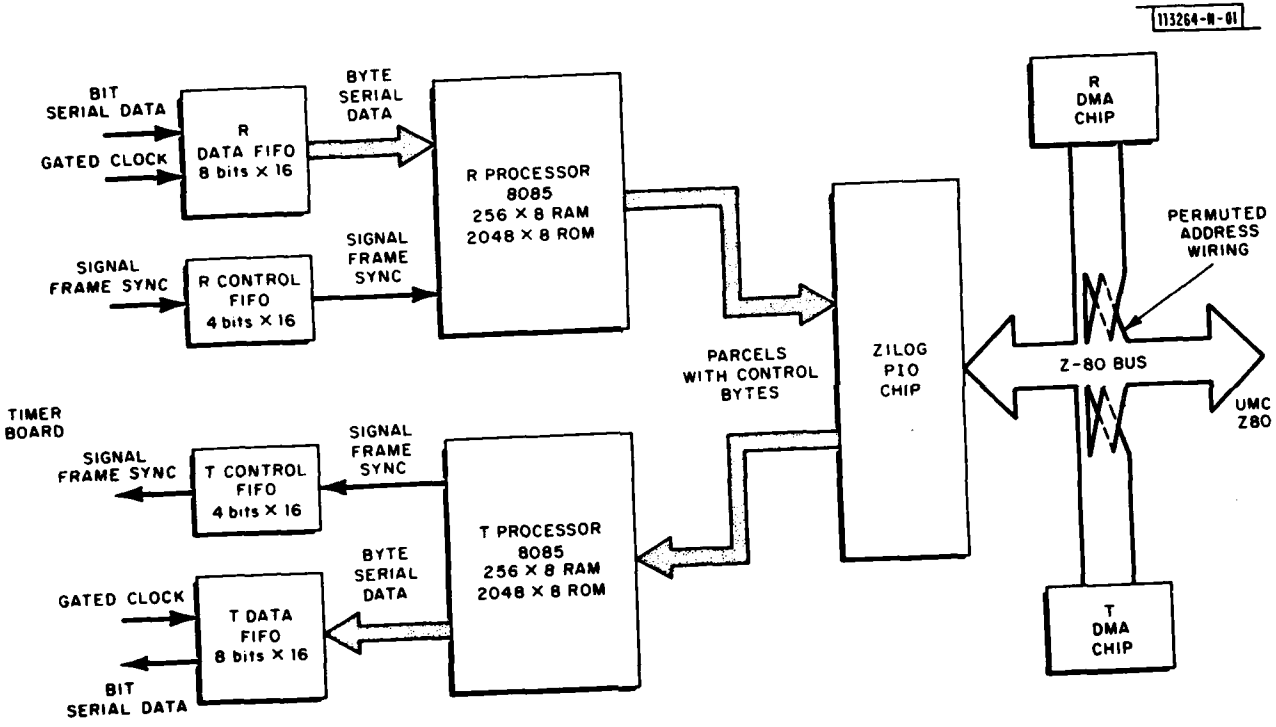


Fig. III-4. The PCI processor board.

2. Packet/Circuit Interface Software

a. Introduction

The Packet/Circuit Interface requires software running on three types of processors: Intel 8085 microprocessors, an Associated Computer Consultants UMC-Z80 microprocessor, and a Digital Equipment PDP-11/44 minicomputer. The two 8085's perform pre- and post-processing, respectively, of the pulse-code-modulated (PCM) speech bytes used with the T1 carrier. This processing includes formation of data blocks and extraction/insertion of E&M signaling. The Z80 essentially provides the signaling bridge between E&M signaling and the packet protocols used in the PSAT, translating in both directions. Finally, the 11/44 runs an internet gateway program, providing a packet-switched connection between the Z80 and the PSAT.

b. 8085 Software Description

The task of the 8085 processors is real-time byte processing, generally including I/O, buffering, and table lookup comparison operations. Each 8085 provides a monodirectional data pipeline. The 8085 preprocessor, or "receive" processor, handles PCM bytes from the channel bank to the Z80; the post-processor or "transmit" processor handles the reverse direction stream.

(1) Receive 8085

Input to the receive 8085 (R8085) consists of some fraction of the channels of the T1 carrier. The proper PCM bytes are gated into hardware FIFO buffers on the 8085 board by special timing circuitry described previously. Processing tasks of the receive 8085 include synchronizing to the channel bank T1, extracting the E&M signaling, silence detection, and formation of data blocks called parcels. Synchronization with the channel bank involves finding the first byte in an E&M sync frame (a 1.5-ms epoch). The channel bank derives this information from the T1 carrier and presents it as an external output. This sync pulse is passed through the FIFOs with the PCM bytes, but in a parallel FIFO channel. Thus, to acquire sync, the R8085 need only look for the sync signal in its exclusive FIFO channel; the data byte in the other FIFO channels is the desired first byte in the frame.

One reason for synchronizing to signaling frames is that an E&M signaling bit replaces the least significant bit of each T1 channel's PCM speech byte during such frames. The R8085 must extract this signaling information. In addition to extracting the signaling, the R8085 must do some time-accumulation, since the opportunity to pass signaling on to the Z80 comes only once per parcel. During the 15 sync frames which constitute a parcel epoch, the processor counts occurrences of both types of signaling bit (for each T1 channel) and uses majority logic to choose which signal level to send to the Z80 in the parcel's control byte. This rejection of signaling information should work acceptably since the standard E&M signaling carries dial pulses, which occur at a maximum rate of ten per second, significantly longer than even the parcel epoch.

Each PCM byte received during a parcel epoch is compared to a silence detection threshold using a lookup table. The table, rather than a direct magnitude compare, is required because the bytes are encoded using the μ255 convention, and speech level does not increase monotonically with encoded value. If any speech sample during the parcel epoch breaks the silence detection threshold, the whole parcel is processed as containing speech. The silence decision is passed on to the Z80 as another of the bits in the parcel's control byte.

The R8085 establishes data blocks for the Z80 by appending one control byte per T1 channel being processed, immediately after sending 180 PCM speech bytes. This creates a natural parcel and Z80 processing epoch of 22.5 ms. As alluded to previously, the parcel control bytes contain: the sync pattern, silence decision bit, and signaling bit.

(2) Transmit 8085

This processor receives parcels from the Z80. In this direction, a parcel for each T1 channel consists of a single control byte, followed by 180 data bytes. Parcel length is 22.5 ms of 64-kbps PCM speech. Actual input to the T8085 consists of a time-multiplexed group of n parcels, where n is a small power of 2 (i.e., 1, 2, 4, or 8), and is the number of T1 channels being processed by the PCI. In the case of four active PCI channels, the present nominal operation mode, the T8085 gets a block consisting of four control bytes, the

first PCM byte from channel 1, the first PCM byte from channel 2, etc., through the 180th byte from channel 4. Block transfer is accomplished by DMA on the Z80 end of the interface and programmed I/O (polling loop) on the 8085 end. Thus, the 8085 gets data in spurts with time-gaps, while the Z80 sets up its controller for subsequent DMA blocks (groups of parcels).

Tasks of the transmit 8085 processor include synchronizing to the Z80's parcel boundaries, time-expanding the M signal samples, generating silence, and generating inband signaling tones. Synchronization with the Z80 is accomplished by searching for a bit pattern known to be resident in the four least significant bits of every control byte. At present, an all-zero pattern is used, since it least resembles the PCM speech samples. The criterion for declaring sync is to see n (i.e., 4) bytes in a row, all containing the sync pattern. Once having acquired sync at power-up, the T8085 should never lose it. The T8085 does confirm sync for every parcel, however, and in the unlikely event of sync loss reenters the initial sync search mode.

Time-expansion of M signal bytes is required, because the parcel epoch (22.5 ms) is 15 times as long as the signaling epoch. So, the T8085 takes each parcel's signaling bit and replicates it in the next 15 E&M signaling frames.

When one PCI calls another, the standard telephony signaling tones, including "ringback" and "busy," are returned to the caller entirely inband, and are actually generated by hardware at the called party's switch or Telephone Office Emulator (TOE). However, when a call is placed for experimental purposes to a phone on a packet-switched network such as the LEXNET,[20] all that is returned to the originating PCI is a ringback or busy packet. Thus, the proper tone must be generated locally and injected into the T1 carrier. What happens is that the Z80 sets one of two special command bits in the control byte for the channel in question. The T8085, upon seeing the command, replaces the PCM speech samples with the tone samples. It has one complete on/off cycle of samples for each category of tone stored in ROM, and continues to send that cycle until commanded to desist.

Output from the T8085 consists of a stream of PCM bytes to a set of FIFO buffer chips, and ultimately to the transmit common unit of the channel bank, which generates the T1 carrier. These PCM bytes have the E&M signaling

bits embedded in the appropriate positions. As with the receive side, only a fraction of the bytes in a fully loaded T1 carrier must be delivered by the 8085. In particular, that amount is now four times 8000 bytes per second, corresponding to four channels of the T1. The bytes are clocked out of the T8085 by hardware on the timing board. I/O to the FIFOs is handled by the 8085 program, not a DMA controller.

The software for both of the 8085's was designed early on in the project in great detail. A "paper coding" exercise, using special symbolic codes but reflecting the execution time required for the actual 8085 instructions, was completed for two reasons: to confirm that the processor would be fast enough to handle its tasks, and to get an idea of the amount of RAM and ROM memory needed for each processor. Given the detailed design, actual coding for the 8085's proceeded very quickly, as did debugging. The code is all in 8085 assembly language, and is written "inline" for speed. Inline code entails repeating groups of statements, rather than setting up loops and taking the cost in overhead for managing the loops. At present, all of the 8085 code is resident on the processor board, in ROM and RAM. During development, the software was run from emulation memory in a Hewlett-Packard development system. The program for the receive 8085 occupies 600 bytes of a single ROM chip whose total capacity is 2048 bytes. Also on that ROM chip is a 256-byte table used for conversion of μ255 samples to absolute magnitudes for silence detection. A single RAM chip, with total capacity of 256 bytes, is used for very temporary storage of the PCM bytes as they transit through the processor. The transmit 8085 code also takes 600 bytes, and its tables, primarily used for storing the ringback and busy tones and silence, another 600 bytes of ROM. This processor also uses one RAM chip for temporary storage of speech bytes.

c. Gateway (PDP-11/44) Software Description

Most of the software in the PDP-11/44 gateway is identical to that developed under DARPA sponsorship[19] to perform internet stream gateway functions. The gateway was designed to support connections to a variety of networks, each connected through UMC-Z80's, and a small software module in the

PDP-11 is required to service each net. In our case, connection between the circuit-switched environment through a PCI, and the satellite net through a packet satellite demand assignment processor (PSAT), was required. The PDP-11 software to serve the PSAT was already available, so only a module to connect to the PCI was required. To minimize the required development effort and maintain compatibility with the packet-switched environment, we designed the PCI to appear to the gateway as equivalent to a special LEXNET connection. The characteristics of this connection include a zero-length local header (since the PCI is circuit switched) and a unique internet address assigned to the PCI. The completed module of PDP-11 software consists of about 24 lines of C (high level) code and a table, occupying in total 700 bytes of PDP memory.

d. Z80 Software Description

The tasks of the PCI Z80 fall in two general categories: conversion of circuit-switched parcels of speech to and from packets, and translation of protocols in both directions between the E&M signaling used in the T1 carrier and the three protocols used in the internet packet environment. These protocols include the end-to-end-level network voice control protocol (NVP) and two transport-level protocols: the standard internet datagram protocol (IP) and an internet stream protocol (ST).

The conversion between speech parcels and packets is programmed totally in Z80 assembly language, and is a comparatively simple task. The only complexities are in synchronizing to the receive 8085's parcel boundaries and in piecing together packets, which may be received out of time-order, into the buffer for playout. Both the E&M signaling and the packet protocols, especially the latter, require rather complex software. The module which implements the packet protocols is programmed entirely in the high-level language C. The two directions of E&M signaling are resident in separate software modules and are programmed in assembly language. To some extent, the E&M modules do simple pre- and post-processing for the C packet protocol module, but they also take some trunk management actions on their own (e.g., seizure of a T1 trunk to service an outgoing call).

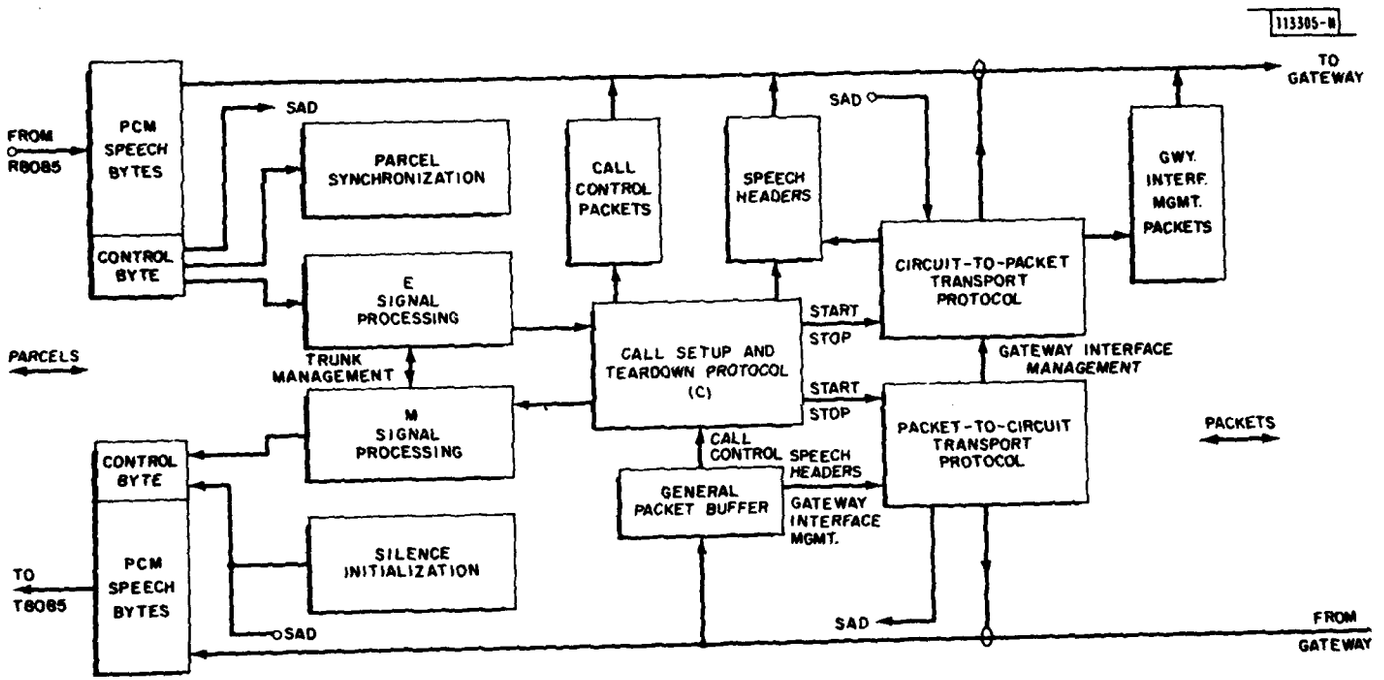


Fig. III-5. PCI Z80 software block diagram.

The C and assembly language code are not linked to each other, and are simultaneously resident in the Z80 by simply overlaying at load time (i.e., two files are loaded). Communication between C and assembly language modules is via common data areas in the Z80 address space.

The detailed discussion of Z80 software which follows is based on two graphical aids: a software module block diagram and a plot of the real-time structure. The block diagram (Fig. III-5) shows the flow of both speech bytes and control through the Z80 processor. It may be viewed as two almost independent sections. The top half or "uplink" section handles speech moving from the channel bank preprocessor ("receive" 8085) through the Z80 to the gateway. Reverse direction flow is depicted on the bottom half or "downlink" section. This terminology is based on the presumption that, on the other side of the gateway, the packets are routed via the PSAT. The real-time structure plot, to be described below, gives an example of the time-ordering of the Z80 activities during a typical parcel epoch.

(1) Software Block Diagram

Refer to Fig. III-5. The uplink section will be described first. In general, the flow of data and control for the uplink is from left to right. The discussion below follows the flow, and, in addition, is time ordered to the extent possible. Large blocks on the figure signify software modules, and the narrow blocks depict buffers and data storage.

On the upper left-most side of Fig. III-5 is a buffer representative of the n pairs of parcel input buffers. Typical double-buffering techniques are used. The number of time slots (or equivalently, channels) being extracted from the T1 carrier and processed by the PCI is n. Speech bytes are placed in the buffers via a Zilog parallel input/output (PIO) chip in conjunction with a direct memory access (DMA) controller. Some of the wires connecting the DMA chip to the Z80 address bus are intentionally attached in nonsequential order, such that the DMA automatically time-demultiplexes the channels. Each buffer is sized to handle a 22.5-ms block of speech samples from one 64-kbps PCM channel, corresponding to a LEXNET "parcel." A parcel contains 180 bytes of data which are followed by one control byte appended by the 8085 preprocessor.

The first uplink software module which is actively invoked deals with acquiring and maintaining parcel synchronization. Parcel synchronization is achieved when the Z80 enables its DMA controller to start at exactly the right instant so that at the completion of the DMA block transfer the buffer contains the 180 speech bytes first, followed by the single control byte. This software module is a state machine whose inputs are fixed sync bit patterns occupying the four least significant bits in each channel's control byte. States include: unsynchronized, time slewing to attain sync, and synchronized. During the unsynchronized state, all bytes in the input buffers are searched for the n copies of the sync pattern. In the parcel epoch immediately after the pattern is located, any offset from the final desired location is compensated. Time slippage is accomplished by setting up the DMA controller to read in a single, short, throw-away block to reach the end of any partial parcel already received. After this, DMA block length reverts to the nominal value. Upon completion of the time base adjustment, the module enters the synchronized state and a flag in the terminal status word is set to indicate to other modules that control and speech bytes are in their proper positions in the buffers, and that these modules may safely start processing the buffer contents. During the synchronized state, the sync module runs in its quick mode, only checking that the sync pattern is where it is expected to be. Any failure to find a sync pattern forces the state machine back into the unsynchronized state. The internal sync flag is reset so that no modules process the incoming bytes until resynchronization has occurred.

The E signal processing module accepts the E&M signaling samples from within each channel's control byte, and uses the samples to drive n concurrent state machines, one for each T1 channel being processed. In some cases of state change, such as a trunk going "off hook," the E signaling module immediately acts in conjunction with its M signaling analogue to complete a trunk management function. The response to such an incoming trunk seizure is for the M signaler to send an acknowledge pulse in the reverse direction on the trunk pair. In other cases of state change, the E signaling module simply changes internal state and waits for further action on the incoming signaling bit stream. An example of this situation would be the receipt of the last

dialing pulse of a phone number digit. The E signaler simply logs the pulse in and waits to accumulate pulses for the next digit. The end of the whole phone number is detected by a "timeout" – an extended period of no signaling activity. Upon entering this "number completed" state, the E signaling module passes the number, and the call control function, on to the call setup and teardown protocol module. The message is placed in the slot in an n-slot queue corresponding to the channel which just changed signaling state. The protocol module, which operates in the background, can accept the signaling message at its leisure.

The call setup and teardown protocol module was designed for implementation in the high-level language C and runs in the background relative to all of the real-time assembly language code. Its task is to run the packet-switched network control protocols. For example, upon receipt of a dialed sequence (phone number) from the E signaler, the call protocol module initiates negotiation with the packet net via the gateway. First it constructs an IP "connect" message in its private call control packet buffer. Then it commands the circuit-to-packet transport module (which has real-time control over all uplink data flow) to send the control packet. This command is indicated by changing a flag byte in the buffer. Ultimately, answering c ll control packets are received, indicating the response of the called party. The way these packets arrive at the call setup and teardown protocol module is described below in the discussion of the downlink section. If the response is positive and it is decided to set up an uplink speech stream, then the call protocol module first creates a "shell" ST header in the block of memory labeled Speech Headers on the figure. Once the shell header is complete, the call protocol module commands that the transport module initiate data flow.

Thenceforth, on a parcel-to-parcel basis, the circuit-to-packet transport protocol module manages ST header contents, including timestamp and checksum, and arranges to send out the complete speech packets. For a given stream, packets are generally sent to the gateway continuously at a rate of one per parcel epoch. The exception occurs in the case where the 8085 preprocessor has detected a full parcel of silence and has set the silence or speech activity detection ("SAD") bit in the channel's control byte. In this case, packet transmission is suppressed.

The following discussion of downlink flow centers around the bottom half of Fig. III-5, and generally progresses right to left on the figure. The philosophy on input from the gateway is for the packet-to-circuit transport protocol module to try to completely clear the gateway's buffer of packets once per parcel epoch. Packet headers are accepted one at a time into a general packet buffer. This buffer has enough slots to store eight of the longest possible call control packets. If the header is that of a speech packet, the speech samples destined for the channel bank (T1 slots) are immediately copied into the appropriate playout buffer, and the slot in the general buffer is cleared for reuse. If the header is that of a gateway interface management message, the body of that message is immediately brought into the general buffer, right below the header, and the message is also acted upon immediately, freeing the buffer slot for reuse. Finally, in the case of call control packets, the body is copied into the general buffer and the whole packet is held for the later use by the call protocol module.

The uplink functions of the call setup and teardown protocol module were already described. One of its highest priority downlink tasks is to clear call control packets from the general packet buffer. Control packets can contain data requiring three general classes of action (reflected in output) by the call protocol module:

(a) Transmission of a return call control packet over the packet net.

(b) Transmission of a signaling message (e.g., "busy") to one of the T1 channels.

(c) Connect/disconnect.

Category 1, a packet handshake, is handled as described previously. If a signaling message is to be sent over the T1 carrier, it is placed in a one-slot queue destined for the M signal processing module. The queue will almost always be empty, since the receiving routine is part of the foreground real-time package, and will act much more quickly than the background level sender. In the case of a call connect, the call protocol module places the (16-bit) backward connection ID in the appropriate slot of a downlink "start" queue.

This queue has one dedicated slot per T1 channel. Actual data flow is initiated by the transport protocol module upon receipt of the first packet with that connection ID. Call disconnects are handled analogously to connects, via a downlink "stop" queue. In this case, it is not necessary to specify the connection ID.

The task of the silence initialization module is simply to initialize the control byte for each parcel with: the (static) sync pattern, a zero signaling bit, and a bit indicating that the parcel contains silence. Later on, the latter two bits may be modified by the M signaler and the transport protocol module, respectively. After placing a speech packet in the playout buffer, the transport protocol module negates the silence bit, indicated by "speech activity detected" (SAD) on the figure. The actions of the silence initializer are non-time crucial since it always stays significantly ahead (i.e., the maximum, 10 parcels) of the read pointer in the playout buffers.

The M signal processing module converts signaling messages originated by the call protocol module and the E signaler to M signaling levels and places the signaling bits in the appropriate position in each parcel's control byte. Like its E signaling twin, it consists of n concurrent state machines, one for each active trunk in the T1 carrier. The M signaling actions are not at all time-crucial, since it also stays well ahead (i.e., four parcels) of the read pointer in the playout buffers (described below) and can thus never deliver a signaling bit too late.

At this point, the buffers for output to the channel bank side 8085 post-processor ("transmit" 8085) must be described. These consist of n circular buffers, each buffer containing an inline chain of 11 parcel slots. Only a single slot of one buffer is depicted in the figure. Format of a parcel is the reverse of the uplink: 180 speech bytes preceded by a control byte. The reason for the size difference over the uplink buffers is that a significant amount of "smoothing" buffering is required in order to compensate for downlink packet arrival time-jitter. As with input from the 8085 preprocessor, the output from the n buffers is time multiplexed automatically by using a DMA controller with some of its address lines crosswired.

(2) Real-Time Structure

The PCI Z80 runs software at three different priority levels: interrupt, foreground, and background. Two devices are allowed to interrupt the Z80 processor: the DMA controllers handling block (parcel) input and output to the 8085 processors. This is the most urgent level of processing, since the DMAs halt at the end of a block, yet the channel bank demands continuous byte streams. First-in-first-out (FIFO) buffers have been placed on the 8085 processor board to smooth over the burstiness in DMA operation, but the FIFOs have finite capacity. Hence it is imperative to quickly reenable the DMA for a new block transfer when it interrupts to signal completion of an old block. The interrupt service routines which perform this task essentially have equal priority, so one cannot interrupt the other; the second arriver simply waits until the first service routine is done and its DMA is reenabled. Basically, the UMC hardware locks out the second interrupt. In terms of DMA idle time, the worst case occurs when one DMA unit tries to interrupt just after the other successfully has captured the processor. This case was analyzed carefully during the design process, and it was possible to code the service routines such that comfortable timing margin was available.

There are two packages of foreground-level processing, one for flow in each direction (uplink and downlink). These packages normally run right after their respective interrupt service routines. On the uplink (or "receive") direction, the philosophy is the standard input-driven one: to perform immediately all processing on the parcels just received. This includes parcel sync, E signaling, and transmission of all packets (speech, call control, and gateway interface management) to the gateway. Similarly in the downlink direction, we attempt to accept and process all packets waiting for the Z80 in the gateway's buffers. This includes carrying the speech packets all the way to the playout buffers. The foreground processing packages are, of course, interruptible by the service routines which set up the DMA controllers. They are designed to have software interlocks such that once one foreground processing package starts to run, the other respects its right to complete, and waits. This is necessary because both packages communicate with the PDP-11 gateway through the same ports, and could potentially interfere.

The background processing consists of a single module of C code which runs the (predominately) packet-switched protocols controlling call setup and teardown. Basically, the background process is given all the spare time in the 22.5-ms parcel epoch, after all the interrupt and foreground level processing is completed. In case there is nothing for the background module to do, it contains a simple wait loop. Typically the speech throughput (foreground) and call management (background) code will complement each other nicely in terms of demand for processor time. For example, when all trunks are busy and the foreground routines require a lot of time for packet processing, the background routine will not receive any new calls and will thus not need much in the way of processor resources.

An example of the real-time structure is shown in the time plot of Fig. III-6. In this case a parcel epoch between receive-side ("R") interrupts is depicted, and the arrival time of the transmit-side interrupt is such that the two do not contend. As described previously, if the T interrupt were to occur earlier, say during the foreground parcel processing, its service routine would run immediately; then the (R direction) parcel processing would be allowed to complete; finally, the (T side) packet processing would occur. Note that the timing of the transmit interrupt will "walk" relative to the receive interrupt for two reasons. At startup, when the Z80 is trying to achieve parcel time sync, some short parcels will be read. This is guaranteed to cause some time slippage, because the output parcel size is to remain fixed. Subsequently,

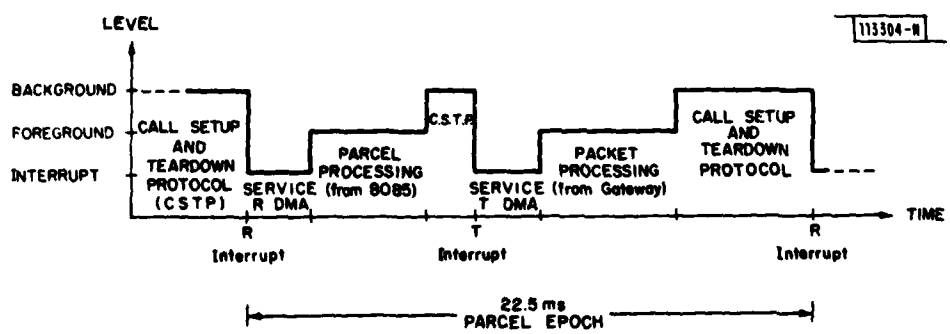


Fig. III-6. PCI Z80 software real-time structure.

if receive and transmit common equipment in the local channel bank are not time locked, the Z80 will also see some relative time drift.

The UMC Z80 is configurable to have one active "page" of memory at a time, consisting of a maximum of 64 K bytes. The operating system and debugger consume 16 K of these bytes. Buffering of the PCM speech parcels takes 4 K and 16 K bytes, respectively, for the receive and transmit directions. The call setup and teardown (C) protocol code requires 12 K bytes, and the assembly language another 8 K. Finally, the common memory used for communication between the C and assembly language code takes 2 K bytes. Thus, at present we have 6 K bytes of Z80 memory to spare.

C. TELEPHONE OFFICE EMULATOR

1. Introduction

The Telephone Office Emulator (TOE) was designed to look and act like a small class 5 telephone office. It will handle up to eight local loops on the subscriber side and six channels of a T1 carrier group on the interoffice trunk side. The TOE provides a switched interface between the subscriber loops and the T1 channels. It has the ability to initiate and receive calls over the trunks using a standard E&M signaling format. A growth capability in the processor was included to enable CCS signaling to be added in the future.

The main purpose of the TOE is to act as a test vehicle and traffic source for the Packet/Circuit Interface (PCI) before connecting the PCI to a T1 carrier of the switched telephone network. The TOE will provide a test bed for verifying the E&M signaling protocols. It will also provide real voice traffic for demonstration experiments. The TOE will also be used to support terrestrial intersite connections (see Fig. III-1) for network routing experiments.

2. Principle of Operation

Figure III-7 shows a very simplified block diagram of a class 5 telephone switching office. On the subscriber loop side the central office must monitor each loop for off-hook or on-hook activity, interpret dialing activity, provide dial tones and ringing (alert) signals. On the interoffice trunk side the central

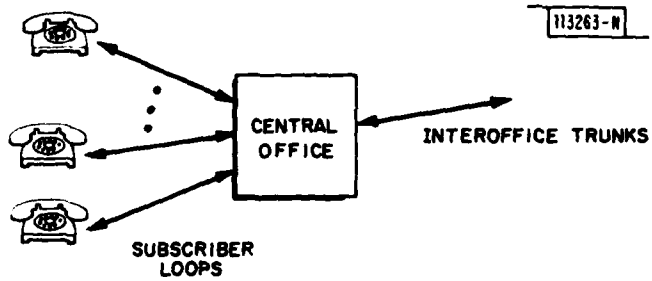


Fig. III-7. Block diagram of class 5 central office.

office must handle the trunk signaling, called number forwarding and receiving, and must provide audible busy and ring signals to the calling central office. The central office also provides switching capability between the inputs. A central office can provide three types of connections. It provides direct connections between subscriber loops (local calls) or between local loops and trunks (inter-office calls). Certain switches also provide trunk-to-trunk (tandem) connections. Because the TOE was designed for a limited set of requirements, it will only provide connections between loops and trunks, although a self-test *looping mode* will be provided which will allow switching between loops.

The TOE is constructed out of components of a Northern Telecom DE3 channel bank with certain modifications. The operation of the TOE is best understood by first reviewing the operation of the unmodified channel bank. Figure III-8 shows a simplified block diagram of the basic channel bank. It consists of two common units – the transmit unit (TU) and the receive unit (RU) – which generate and receive the T1 carrier. The channel bank also contains a separate channel unit (CU) for each subscriber loop. Analog voice signals and loop signaling bits are transmitted between the CUs and the TU and RU by common buses. The analog signals travel on Pulse Amplitude Modulated buses (TPAM and RPAM). The signaling bits use a pair of digital buses, RSAB and TSA/TSB. In addition, the TU and RU each generate a set of 24 timing signals (RCCn- and TCCn-) for the individual channel units. When the receive timing signal for a particular channel unit is active, that CU accepts an analog sample from the RPAM bus and on signaling frames, a signaling bit from the

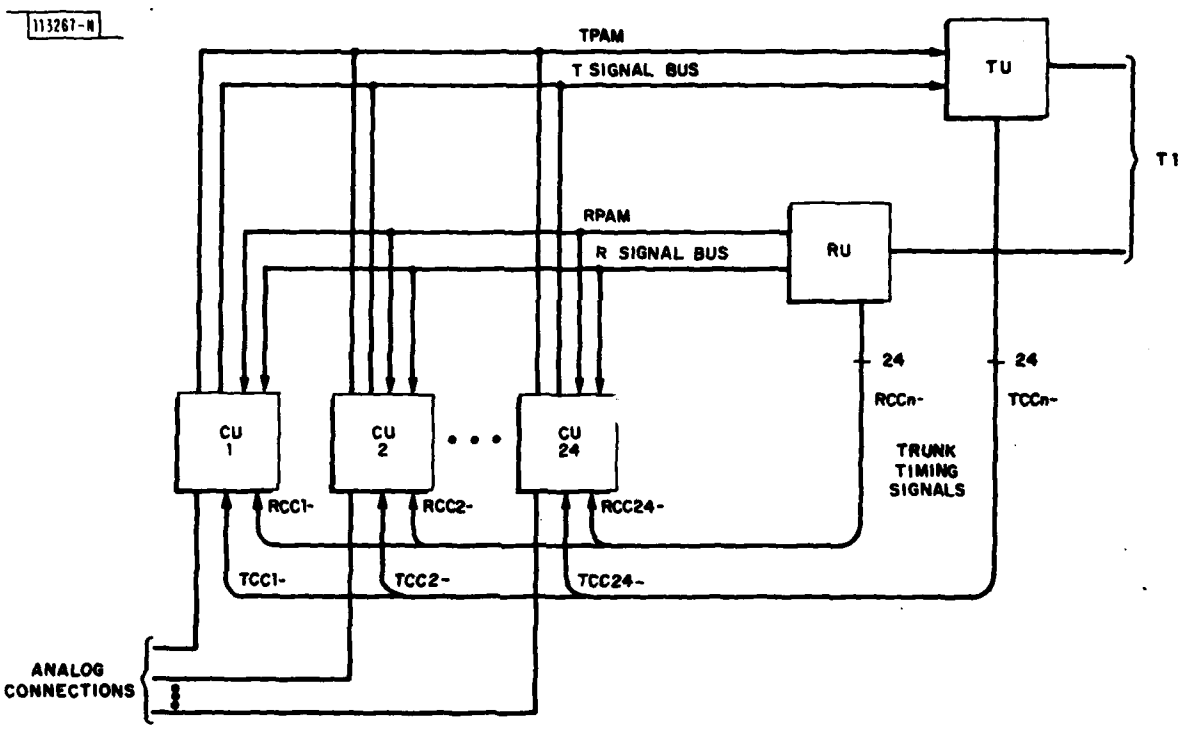


Fig. III-8. Normal channel bank operation.

RSAB bus. When the transmit timing bit for a particular CU is active, the CU outputs analog voice samples on the TPAM bus and signaling bits on the TSA and TSB signaling bus. The TU discards signaling bits that are sent during non-signaling frames.

The structure of the TOE is basically the same but with a few important changes shown in Fig. III-9. First, the signaling bus has been broken. Separate registers are used for the signaling bits on the loop side and on the trunk side. Second, the timing signals RCCn- and TCCn- are interrupted by a multiplex switch. Third, a tone generator has been added on the RPAM and TPAM buses. Th additions are all under the control of an 8085 microprocessor. The functions of these additions are described below.

Signaling Registers

In the unmodified channel unit, the signaling is transparent from one central office to the next. Whatever signaling inputs are present at the input of a channel unit will be delivered to the output of the channel unit in the same slot at the remote end of the T1 carrier. In the TOE, the situation is different. The signaling bits from the subscriber loop side are connected to the CU Signaling Register. These signaling bits are interpreted by the 8085 in a manner appropriate for a local subscriber loop (off-hook, dial pulse, etc.). The signal bits from the TU and RU are connected to the Trunk Signaling Register. The 8085 handles these as interoffice E&M trunk signals. The CPU program handles the interface protocol between the two sides.

Switching

The switching function is also controlled by the CPU. The timing lines for each CU come from the outputs of selectors which select one of six RCCn- and TCCn- lines from the TU and RU. This enables any CU to be connected to any T1 slot.

Tone Generator

The tone generator card generates tones which can be sampled onto the analog buses. This card can provide either a busy signal or a ringing tone

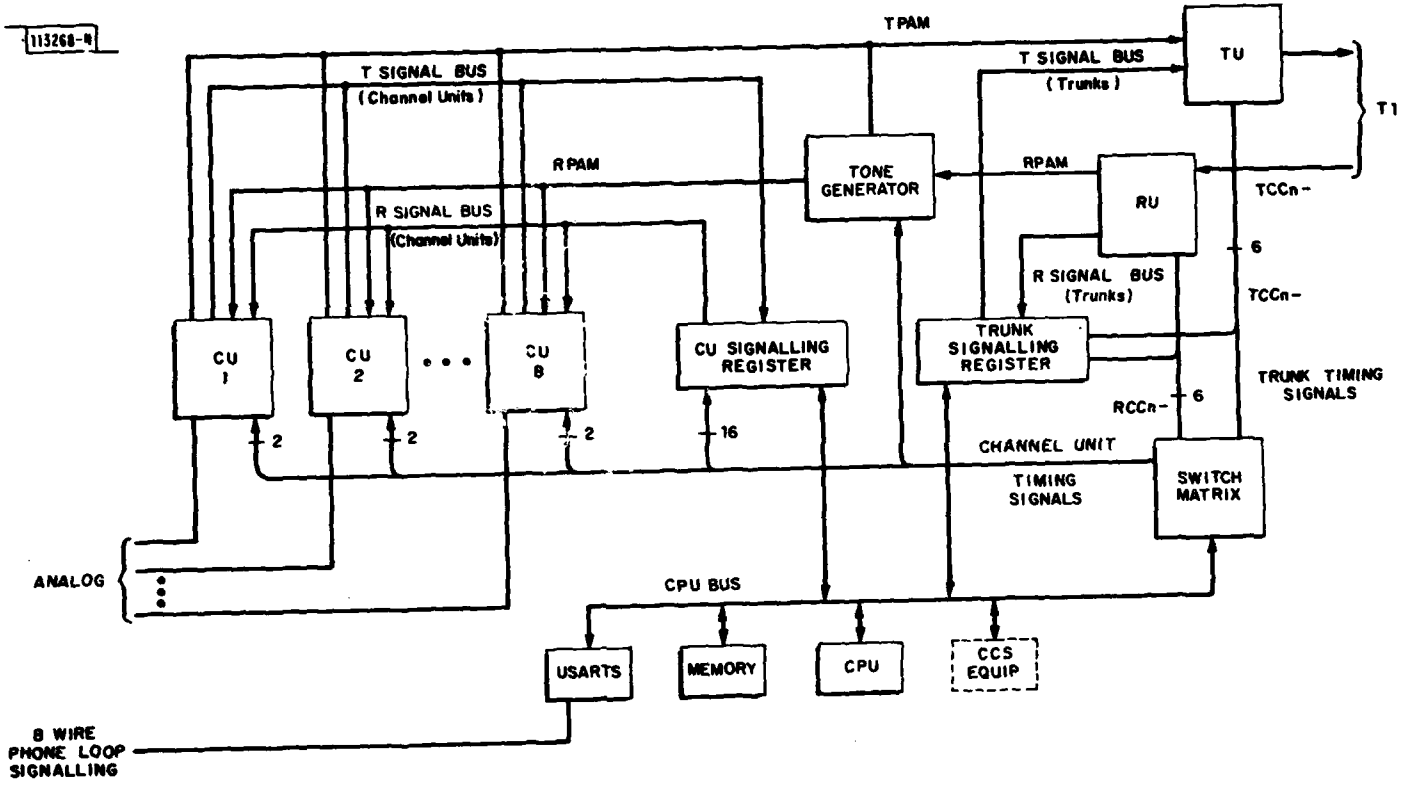


Fig. III-9. TOE block diagram.

to the TPAM bus. These tones are sent to the remote end of the T1 connection to indicate that the called phone at the local end was busy or was ringing. A dial tone can be inserted on the RPAM bus in any set of CU time slots. This provides the dial tone to phones on the local loops.

3. Channel Unit Options

Two types of channel units will be used to support different types of local telephones connected to the TOE. The 4-wire E&M CUs are designed for 4-wire analog trunks. In the TOE they will be used with the "8-wire" phones developed for use with packet voice terminals.[20] These phones have separate send and receive analog circuits and signaling channels. The signaling channel is via an RS-232 connection. In TOE channels equipped with this combination, all signaling is over the RS-232 connection. This signaling channel also controls the ringing and dial tones which are contained in the phone. This combination also provides a 4-wire audio connection right to the user. This avoids having to deal with echo suppression or echo cancellation. The second type of channel unit is the FXS which was designed to provide the subscriber end of a foreign exchange circuit. This unit is normally used in conjunction with the FXO channel unit at the office end. In the TOE, the FXS channel unit is used with a standard 2-wire rotary-dial phone. The CU contains a hybrid transformer for the 2-wire to 4-wire conversion. Switches on the channel unit allow it to be balanced to match the local loop. The channel unit will also sense the hook switch state and dial pulse interruption and convert this information to signaling bits which are sent to the CPU via the CU Signaling Registers. The CU can also apply a ringing voltage to the loop on command of the CPU.

4. Front Panel

A front panel display has been added to the channel bank to allow the trunk and loop status to be examined. Single LEDs are used to show the state of the trunks and subscriber loop. Digital displays show the switch connections and the progress of the dialing. A keyboard input is used to select different displays and provide a means for any other required program inputs.

5. Construction

The prototype of the TOE is constructed in the cardrack of the channel bank. The printed-circuit backplane of the channel bank has been redesigned for the TOE application. Five (and possibly six) new cards will be added. The CPU card contains the 8085 CPU, memory, timing, and control of the display panel. The trunk card contains the switching elements for the TU and RU timing, and registers for the trunk signaling bits. The USART card handles the signaling for the 8-wire phones. The FXS card handles the signaling for the 2-wire phones. Provision has been made for an added memory should the program grow beyond what can be accommodated on the CPU card. The tone generator card provides the necessary analog signaling tones. These cards all reside in the channel bank in place of CU cards. The power supplies and display panel are mounted on top of the channel bank.

6. Status

One prototype TOE unit has been constructed, tested, and shown to perform its basic functions correctly. Testing of the TOE in combination with the PCI is under way.

D. INTERNET PACKET GATEWAY

The Internet Packet Gateway (IPG), formerly called "Applique D," is a gateway for DoD standard Internet Protocol (IP) data packets. When delivered to DCEC, it will function as a gateway between the wideband satellite network of EISN and the Experimental Data Network (EDN) at DCEC. It makes use of the same hardware as the PCI gateway (a PDP-11/44 computer and two UMC-Z80 interface cards) with the addition of an added UMC-Z80 and a distant-host interface card to connect to the EDN. Concurrent operation of the PCI and the EDN interface is possible, with the PDP-11/44 then becoming a shared resource for voice and data traffic.

Both the PCI and the IPG make use of PDP-11 and Z80 software already developed for the DARPA portion of the wideband satellite experiment. This software handles IP core gateway functions and interfaces to the PSAT. It

has been under development throughout FY 81 and has been demonstrated successfully in loop tests using speech packets generated in packet voice terminals connected to a local cable net (LEXNET) at Lincoln Laboratory and the Information Sciences Institute (ISI) in California.

The Z80 software required to interface to the EDN has been written and tested using an ARPANET port at Lincoln. The EDN and ARPANET use the same HOST-to-IMP protocols as specified by BBN Report No. 1822.[21] Testing of this interface has been limited to narrowband speech experiments between Lincoln and ISI using the ARPANET.

Testing of the IPG with loads more typical of data communications will occur when the gateway is connected to EDN and hosts with DoD standard Transmission Control Protocol (TCP) capability are available as test generators. We do not have such hosts currently available at Lincoln.

REFERENCES

1. Network Speech Systems Technology Annual Report, Lincoln Laboratory, M.I.T. (30 September 1980), DTIC AD-A097998/9.
2. J. W. Forgie et al., "Voice Conferencing Technology Program – Final Report," Lincoln Laboratory, M.I.T. (31 March 1979), DDC AD-A074498/7.
3. R. P. Lippmann, "Routing in Circuit and Packet Switched Networks: An Annotated Bibliography," Technical Report 585, Lincoln Laboratory, M.I.T. (14 September 1981), DTIC AD-A106155.
4. "DSN Alternative Networks," Defense Communications Engineering Working Paper (Draft) (May 1981).
5. S. Katz, "Alternate Routing For Nonhierarchical Communication Networks," Proc. IEEE Natl. Telecommunications Conf., December 1972, pp. 9A.1-9A.8.
6. J. W. Gorgas, "AUTOVON: Switching Network for Global Defense," Bell Lab. Rec. 46, 106-111 (1968).
7. H. M. Heggestad, "A Flexible Routing Algorithm for the Future Defense Switched Network," Proc. IEEE Natl. Telecommunications Conf., Houston, Texas, 30 November – 4 December 1980, Vol. 1, pp. 74.3.1-74.3.5, DTIC AD-A100999/2.
8. J. H. Weber, "A Simulation Study of Routing and Control in Communications Networks," Bell Syst. Tech. J. 43, 2639-2676 (1964).
9. J. McQuillan, I. Richer, and E. C. Rosen, "The New Routing Algorithm for the ARPANET," IEEE Trans. Commun. COM-28, 711-719 (1980).
10. A. Segall, "Optimal Distributed Routing for Virtual Line-Switched Data Networks," IEEE Trans. Commun. COM-27, 201-209 (1979).
11. J. M. McQuillan, G. Falk, and I. Richer, "A Review of the Development and Performance of the ARPANET Routing Algorithm," IEEE Trans. Commun. COM-26, 1802-1810 (1978).
12. M. J. Fischer, D. A. Garbin, T. C. Harris, and J. E. Knepley, "Large Scale Communication Networks – Design and Analysis," OMEGA: Int. J. Mgmnt. Sci. 6, 331-340 (1978).
13. G. Ludwig and R. Roy, "Saturation Routing Network Limits," Proc. IEEE 65, 1353-1362 (1977).

14. G. W. Bernas and D. M. Grieco, "A Comparison of Routing Techniques for Tactical Circuit-Switching Networks," ICC '78 Conference Record, Toronto, Canada, 4-7 June 1978, Vol. 2, pp. 23.5.1-23.5.5.

15. C. N. Shearer, "Modified Forward Routing for CONUS/CSN AUTOVON," Defense Communications Engineering Center Technical Report 26-75 (September 1975).

16. P. M. Lin, B. J. Leon, and C. R. Stewart, "Analysis of Circuit-Switched Networks Employing Originating-Office Control with Spill-Forward," IEEE Trans. Commun. COM-26, 754-765 (1978).

17. W. S. Chan, "Recursive Algorithms for Computing End-to-End Blocking in a Network with Arbitrary Routing Plan," IEEE Trans. Commun. COM-28, 153-164 (1980).

18. M. D. Gaudreau, "Recursive Formulas for the Calculation of Point-to-Point Congestion," *IEEE Trans. Commun.* COM-28, 313-316 (1980).

19. Information Processing Techniques Program Semiannual Technical Summary Vol. 1: Packet Speech Systems Technology, Lincoln Laboratory, M.I.T. (31 March 1980), DDC AD-A092113/0.

20. G. C. O'Leary, P. E. Blankenship, J. Tierney, and J. A. Feldman, "A Modular Approach to Packet Voice Terminal Hardware Design," AFIPS – Conference Proceedings, Vol. 50 (AFIPS Press, Arlington, Virginia, 1981), pp. 183-189, DTIC AD-A105625.

21. Bolt Beranek and Newman, Inc., "Specification for the Interconnection of a Host and an IMP," Report 1822 (May 1978) (revised).

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ESD-TR-81-336	2. GOVT ACCESSION NO. AD-A112 718	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* Network Speech Systems Technology Program		5. TYPE OF REPORT & PERIOD COVERED Annual Report 1 October 1980 – 30 September 1981
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR*(s)* Clifford J. Weinstein		8. CONTRACT OR GRANT NUMBER*(s)* F19628-80-C-0002
9. PERFORMING ORGANIZATION NAME AND ADDRESS Lincoln Laboratory, M.I.T. P.O. Box 73 Lexington, MA 02173-0073		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Program Element No. 33126K
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Communications Agency 8th Street & So. Courthouse Road Arlington, VA 22204		12. REPORT DATE 30 September 1981
		13. NUMBER OF PAGES 80
14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)* Electronic Systems Division Hanscom AFB, MA 01731		15. SECURITY CLASS. *(of this report)* Unclassified
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

None

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

network speech processing
secure voice conferencing
speech algorithms
integrated networks
packetized speech
human factors
speech encoding
teleconferencing

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

This report documents work performed during FY 1981 on the DCA-sponsored Network Speech Systems Technology Program. The two areas of work reported are: (1) communication system studies in support of the evolving Defense Switched Network (DSN) and (2) design and implementation of satellite/terrestrial interfaces for the Experimental Integrated Switched Network (EISN). The system studies focus on the development and evaluation of economical and endurable network routing procedures. Satellite/terrestrial interface development includes circuit-switched and packet-switched connections to the experimental wideband satellite network. Efforts in planning and coordination of EISN experiments are reported in detail in a separate EISN Experiment Plan.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*